ILVTY RY:N WEB-SIVUSTO

Jäsensivusto ja tietoturva

Jani Laitinen

Opinnäytetyö

Ammattikorkeakoulututkinto

Savonia
ammattikorkeakoulu

SAVONIA-AMMATTIKORKEAKOULU **OPINNÄYTETYÖ**
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikka	
Työn tekijä(t) Jani Laitinen	
Työn nimi ILVTY RY:n web-sivusto - jäsensivusto ja tietoturva	
Päiväys 15.5.2012	Sivumäärä/Liitteet 59/12
Ohjaaja(t) Lehtori Veijo Pitkänen	
Toimeksiantaja/Yhteistyökumppani(t) Ilmavoimien lennonvarmennusteknillinen yhdistys	

Tiivistelmä

Tämän päättötyön aiheena oli toteuttaa Ilmavoimien lennonvarmennusteknilliselle yhdistykselle internet-sivusto yhdistyksen toiminnan esittämiseksi kohdeyleisölle ja jäsenkannan ylläpitäjien tarpeisiin.

Vaatimuksena oli toteuttaa helppokäyttöinen selaimella käytettävä tietokantapohjainen sovellus tietojen päivittämiseen ja arkistointiin. Ratkaisun tulisi perustua huokeisiin tai ilmaisiin ohjelmistoihin. Mahdollisen julkiseen verkkoon sijoittamisen vuoksi tulisi sivuston ja tuotantoympäristön tietoturvallisuuteen kiinnittää erityistä huomiota. Työ jakaantuu kahteen osaan, joista ensimmäisessä pohdin tietoturvaa yleisesti sekä itse sovelluksen määrittelyä. Toisessa osassa toteutan sovelluksen toiminnot osittain.

Tavoitteena oli luoda jäsen-, hallitus- ja kokoustietojen ylläpitoon sekä tapahtumien kirjaamiseen liittyvät toiminnot. Lisäksi sivuston tietokantaan tulisi luoda mahdollisuus kirjata jäsenmaksujen suoritukset. Toteutin työssä pääosin kaikki toiminnot, mutta sivuston ulkoasu jää jatkotyöskentelyn varaan. Työssä saadut tulokset antavat hyvän pohjan sivuston loppuun toteuttamiselle ja tukevat päätöksentekoa lopullisesta toteutusympäristöstä ja käyttöönotosta.

Kehitysympäristönä käytin LAMP-alustaa, jossa käyttöjärjestelmänä on Linux, tietokantana MySQL, www-palvelimena Apache ja sivuston toiminnallisuudet luotiin PHP-kielellä. Esittelin työssä sivuston rakenteen ja huomioon otetut turvallisuuteen liittyvät asiat. Käsittelin työssä myös edellä mainittujen ohjelmistojen turvallisuutta ja tietoturvaa yleisesti.

Avainsanat
LAMP, Apache, Linux, MySQL, PHP

julkinen

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Information Technology			
Author(s) Jani Laitinen			
Title of Thesis Web pages for ILVTY RY - Application and data protection			
Date	15.5.2012	Pages/Appendices	59/12
Supervisor(s) Lehtori Veijo Pitkänen			
Project/Partners Ilmavoimien lennonvarmennusteknillinen yhdistys			

Abstract

The aim of this thesis was to develop web pages for the ILVTY registered association to provide information for their members in an improved manner and help the board of directors to maintain membership register. ILVTY RY is a non-profit registered association which tries to attend the benefits of the technical personnel in the air force excluding aeronautics. Aeronautics has their own association.

The application should be based on free or budget friendly software. It should be operated by web browser and it should use database for data storing. One should have to pay extra attention for data security and protection because of the character of the data. This thesis is divided into two sections - the first part mainly considers data protection and the definition of the application itself. In the second part the application is implemented.

Most of the functionality of the application was carried out but the overall appearance needs improving. The results of this project will give a good groundwork for finishing web pages later on. This thesis will help satisfyingly the board of ILVTY RY to decide whether to implement web pages or not and in which way if they are to be executed.

This project was carried out by using open source software; Linux, Apache and MySQL. The functionality in web pages was implemented by using PHP and AJAX -techniques.

Keywords
LAMP, Apache, Linux, MySQL, PHP

public

SISÄLTÖ

LIITTEET

TERMIT JA LYHENTEET

Apache	Avoimeen lähdekoodiin perustuva ilmainen HTTP-palvelinohjelma.
Avoin lähdekoodi	Avoimella lähdekoodilla (open source) viitataan ohjelmiin, joita käyttäjä voi itse korjata tai parannella koska ohjelmakoodi on yleisesti saatavilla. Tällaiset ohjelmat ovat usein täysin ilmaisia.
DMZ	Verkon alue, joka sisältää internetistä tai muusta verkon turvattomammasta alueesta käsin käytettäviä laitteita. DMZ-verkkoon (demilitarized zone) sijoitetaan tavallisesti internet-käyttöön tarkoitetut palvelimet (www, ulkoinen DNS) ja yhdyskäytävät (VAHTI Sisäverkko-ohje, luonnos 9.3.2010).
Dynaaminen websivu	Dynaaminen sivu luodaan vasta, kun web-selain pyytää sitä. Selaimen hakupyyntö käynnistää palvelimella toimintoja, joiden tuloksena syntyy uusi verkkosivu.
GNU	Projekti, jonka tavoitteena oli kehittää täydellinen vapaa käyttöjärjestelmä, joka koostuisi ainoastaan vapaista ohjelmista. GNU:sta on sittemmin kehittynyt monia muita vapaan lisenssin projekteja.
GPL-lisenssi	GPL-lisenssi takaa käyttäjälle oikeuden kopioida, muuttaa ja jakaa edelleen ohjelmia sekä niiden lähdekoodia.
Http	Hypertext Transfer Protocol. Protokolla, jota selaimet ja WWW-palvelimet käyttävät tiedonsiirtoon.
Klausuuli	Erityisesti rajoittava tai selittävä lisä- tai ehtopykälä.

LAMP	Kokoelma avoimeen lähdekoodiin perustuvia ohjelmia (Linux, Apache, MySQL, PHP), jotka yhdessä muodostavat WWW-palvelimen ja jonka avulla voidaan suorittaa dynaamisia websivuja.
Linux	Avoimen lähdekoodin käyttöjärjestelmä, josta on olemassa useita erilaisia jakeluversioita.
MySQL	SQL-tietokannan hallintajärjestelmä.
Palvelin	Tietokone, jossa suoritetaan palvelinohjelmistoa, jonka tehtävänä on tarjota erilaisia palveluja muille ohjelmille joko tietokoneverkon kautta tai paikallisesti samassa tietokoneessa.
PHP	Ohjelmointikieli, jota käytetään erityisesti web-palvelinympäristössä dynaamisten verkkosivujen luonnissa.
Tietokanta	Kokoelma tietoja, joilla on yhteys toisiinsa.
Tietue	Tietokannassa olevan taulun yksi rivi.
Webmin	Selaimella toimiva Perl-pohjainen hallintatyökalu unix-pohjaisten järjestelmien palvelintoiminnan hallintaan. Webmin toimii palvelimena oletuksena TCP-portissa 10000.

1 JOHDANTO

Tämän työn tarkoituksena oli luoda Ilmavoimien lennonvarmennusteknilliselle yhdistykselle (ILVTY) www-sivusto työpaikan sisäiseen intranet-verkkoon tai vaihtoehtoisesti julkiseen internettiin. Sivuston tulisi hyödyntää palvelimella sijaitsevaa tietokantaa jäsenrekisterin ylläpitämiseksi. ILVTY ry on nuori yhdistys, joka ajaa Ilmavoimien teknillisen henkilöstön etuja. Jäseneksi kelpuutetaan sotilasvirassa palvelevat henkilöt, jotka työskentelevät lennonvarmistukseen liittyvissä tehtävissä. Jäsenten lukumäärä on tällä hetkellä noin 300 henkilöä.

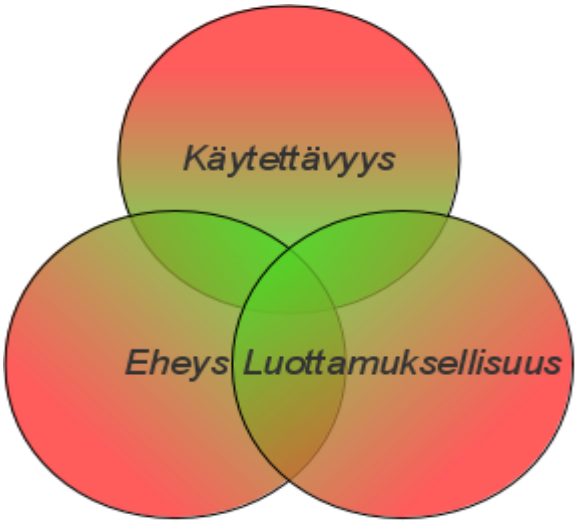
Sivuston mahdollisen julkiseen verkkoon siirtymisen vuoksi tuli työssä kiinnittää huomiota sivuston ja tuotantoympäristön turvallisuuteen. Työn alkumäärittelyissä oli tavoitteena saada helppokäyttöinen tietokantapohjainen sovellus tietojen päivittämiseen ja arkistointiin. Lisäksi ratkaisun tuli perustua huokeisiin tai ilmaisiin ohjelmistoihin.

Työn tekniseksi alustaksi valittiin seuraava kokonaisuus; Linux, Apache ja MySQL sekä toiminnallisuuden luomiseen skriptikieli PHP. Työssä esiteltiin sivuston ja alustan rakenne ja huomioon otetut turvallisuuteen nojautuneet ratkaisut. Työssä käsiteltiin erikseen valittujen ohjelmistojen turvallisuutta kuin myös verkkoturvallisuutta yleensä. Lopuksi sivusto toteutettiin osittain ja toimivuus testattiin.

2 TIETOTURVALLISUUS

Tietoturvaa ja turvallisuutta yleisesti pohdittaessa voidaan todeta, ettei täydellisen murtovarmaa sovellusta voida tehdä. Lisäksi käytettävyys mahdollisesti heikkenee turvallisuutta parannettaessa. Tämän vuoksi valitut ratkaisut ovat pitkälti tasapainottelua turvallisuuden ja käytettävyyden välillä. Hyvin tärkeää on alussa miettiä mitkä ovat uhkatekijät ja kuinka salaisena käsiteltävää tietoa pidetään. Sivuston tarkoituksena on helpottaa jäsenrekisterin ylläpitoa, jolloin joutunemme tarkistamaan henkilötietolaista mitä vaateita tämä tuo mukanaan. *"Rekisterinpitäjän on toteutettava tarpeelliset tekniset ja organisatoriset toimenpiteet henkilötietojen suojaamiseksi asiattomalta pääsyltä tietoihin ja vahingossa tai laittomasti tapahtuvalta tietojen hävittämiseltä, muuttamiselta, luovuttamiselta, siirtämiseltä taikka muulta laittomalta käsittelyltä"* (Henkilötietolaki 32 §, Tietojen suojaaminen). Edellinen lain lause antaa hyvän lähtötason millä vakavuudella sivuston toteutusta on mietittävä.

Perinteisessä tiedon arvoon perustuvassa määritelmässä tietoturvallisuus koostuu kolmesta osatekijästä, jotka ovat luottamuksellisuus, käytettävyys ja eheys (Hakala, Vainio & Vuorinen 2006, 4). Luottamuksellisuudella tarkoitetaan sitä, että tieto on vain niillä, joille tieto kuuluu. Se suojaa jonkin asian omistusoikeutta ja yksityisyyttä. Käytettävyydellä tarkoitetaan sitä, että tieto on aina käytettävissä niillä, jotka sitä tarvitsevat ja ovat siihen oikeutettuja. Eheydellä tarkoitetaan tiedon muuttumattomuutta sitä siirrettäessä tai säilytettäessä (Kaario 2002, 293). Tietojen luottamuksellisuutta, eheyttä ja käytettävyyttä turvataan laitteisto- ja ohjelmistovikojen, luonnontapahtumien sekä tahallisten, tuottamuksellisten tai tapaturmaisten tekojen aiheuttamilta uhilta ja vahingoilta (www.yliopistojentt.fi/VAHTICD/Sivusto/kayttajan_ohje/011_johdanto1 luettu 10.02.2010).



KUVIO 1. Tietoturvallisuuden osatekijät

Nykyisin monissa yhteyksissä pidetään klassista määritelmää riittämättömänä, joten määritelmää on laajennettu vähintään kahdella osatekijällä; kiistämättömyys ja pääsynvalvonta (Hakala ym. 2006, 5). Osapuolten todentaminen ja tapahtuman kiistämättömyys ovat erityisen tärkeitä silloin, kun järjestelmän käyttäjät tulee pystyä tunnistamaan esimerkiksi käytettäessä sähköisiä asiointipalveluita tai etätyötä tehtäessä (www.yliopistojentt.fi/VAHTICD/Sivusto/kayttajan_ohje/011_johdanto1 luettu 10.02.2010).

Yleisesti tietoturvallisuuden osatekijät pilkotaan helpommin ymmärrettäviin ja käsiteltäviin osiin kuten lähteenä käyttämässäni Tietoturvallisuuden käsikirjassa mainitut seuraavat osa-alueet: hallinnollinen, fyysinen, henkilö-, tietoaineisto-, ohjelmisto-, laitteisto- ja tietoliikenneturvallisuus. Tulen itse käyttämään edellä mainittua jaottelua. Turvallisuutta pohdittaessa voidaan ohjenuoraksi ottaa myös esimerkiksi Valtionhallinnon Vahti-tietoturvaohje, joka jakaa turvallisuuden käsitteineen ja määrittelyineen vieläkin pienempiin osa-alueisiin. Lisää suuntalinjoja tietoturvallisuuteen saa kansallisesta tietoturvakriteeristöstä (KATAKRI).

2.1 Hallinnollinen turvallisuus

Hallinnollinen turvallisuus käsittää tietoturvan kehittämisen ja johtamisen sekä yhteydenpidon muihin turvallisuudesta vastaaviin elimiin ja viranomaisiin. Lisäksi tärkeässä asemassa on lainsäädännön ja lisenssi- sekä palvelusopimusten vaikutusten arviointi tietoturvakäytäntöihin. Hallinnollisen turvallisuuden ylläpito kuuluu yleensä tietohallin-

non tehtäviin (Hakala ym. 2006, 11). Hallinnolliseen tietoturvallisuuteen kuuluu henkilöstön tehtävien ja vastuiden määrittely. Tietohallinnon tulee luoda yrityksen toimintaan soveltuva tietoturvapolitiikka sekä -ohjeistus. Tämä kohta on enemmissä määrin otettava huomioon, jos sivusto sijoitetaan esimerkiksi ulkopuolisen palveluntarjoajan palvelimelle. Tällöin on arvioitava palveluntarjoajan tietohallinnon tasoa. Muutoin on yhdistyksen nimettävä itselleen tietohallinnosta vastaava henkilö, joka tarpeellisella panostuksella hoitaa hallinnolliseen turvallisuuteen liittyviä tehtäviä..

2.2 Fyysinen turvallisuus

Fyysisellä turvallisuudella käsitetään laitteiden kuten palvelimien suojaamista fyysisiltä uhkilta kuten ilkivallalta ja murroilta. Helposti unohdetaan myös ympäristön aiheuttamat uhat kuten esimerkiksi mahdollisuus vesi- ja palovahinkoihin tai sähkö- ja lämmitysjärjestelmien toimintahäiriöihin. Tosin jälkimmäisen sijasta tärkeämpää lienee pohtia riittävää jäähdytystä. Helpointa on, jos palvelin voitaisiin sijoittaa jonkin suuremman yrityksen tiloihin, jolloin pääsynvalvonta sekä muu fyysinen turvallisuus on todennäköisesti jo valmiiksi hoidettu hyvin. Mikäli yhdistyksen jäsenten työnantaja ei halunne tarjota palvelimelle tiloja, tullee ulkopuolisen palveluntarjoajan tilat ja pääsynvalvonnan järjestelyt arvioida tarkasti. Vähimmäisvaatimuksena voitaisiin pitää laitteiden toiminnan turvaamista UPS-järjestelmillä, tilojen suojaamista sammutusjärjestelmillä, kameravalvonnalla ja tilat tulisi olla mielellään varustettu elektronisella kulunvalvonnalla. (VAHTI 1/2002.)

2.3 Henkilöturvallisuus

Henkilöstöturvallisuudessa on kyse henkilöstöstä aiheutuvien riskien hallinnasta ja henkilöturvallisuudessa ihmisiin kohdistuvien riskien hallinnasta (VAHTI 2/2008). Käsittelen tässä kuitenkin ihmisistä aiheutuvia riskejä henkilöturvallisuuden nimikkeen alla. Henkilöturvallisuuteen kuuluvat ne toimet, joilla varmistetaan tietojärjestelmän käyttäjien toimintakyky sekä rajataan heidän mahdollisuuksiaan käyttää tietoja ja tietojärjestelmiä (Hakala ym. 2006, 11). Tähän lukeutuu tietojärjestelmiin liittyvä kouluttaminen, sijaisuusjärjestelyt ja -määrittelyt, tietojen saantiin ja tietojärjestelmiin liittyvien oikeuksien ja vastuiden määrittäminen. Kriittisten tehtävien osalta tehtävänkuvaukset tulisi dokumentoida riittävän hyvin, jotta sijainen pystyy ohjeiden avulla suorittamaan tehtävän. Henkilöstöturvallisuuden osalta tulisi jokaisen muistaa, että hän on itse vastuussa tehtäviinsä liittyvästä tietoturvallisuudesta. Tulee myös muistaa, että suurin osa tietorikoksista tapahtuu yrityksen sisällä työntekijän tai entisen työntekijän toimesta. Tämän vuoksi tulee jokaisen yrityksen kiinnittää erityistä huomiota myös

henkilöstöpolitiikkaan sekä työilmapiiriin. Tyytyväinen työntekijä pysyy työssään ja vähentää henkilöongelmien ilmaantumista.

2.4 Tietoaineisto- ja ohjelmistoturvallisuus

Tietoaineistoturvallisuuteen kuuluvat tietojen säilyttämiseen, varmistamiseen ja palauttamiseen sekä tuhoamiseen liittyvät toimet (Hakala ym. 2006, 11). Aineistoon luetaan myös paperiset tulosteet ja asiakirjat. Ohjelmistoturvallisuuteen liittyvät ohjelmistoihin liittyvät seikat kuten testaukset ja koekäytöt sekä ohjelmistoversioiden ja lisenssien hallinta.

2.5 Laitteistoturvallisuus

Osa-alueeseen liittyvät tietokoneiden ja muiden tietojärjestelmään kytkettyjen laitteiden tarkoituksenmukainen mitoitus, toiminnan testaus, huollon järjestäminen sekä varautuminen laitteiden kulumiseen ja vanhentumiseen. Lisäksi tulee ottaa huomioon laitteiden käytöstä aiheutuvien vaaratekijöiden arviointi ja minimointi.

2.6 Tietoliikenneturvallisuus

Tietoliikenneturvallisuudella tarkoitetaan tiedonsiirtoratkaisujen ja muiden viestintäjärjestelmien turvallisuudesta huolehtimista. Tässä yhteydessä käytän myös termiä verkkoturvallisuus käsittäen TCP/IP-verkkoon liittyviä tietoturvauhkia. TCP/IP-verkoissa lähes kaikki tieto kulkee salaamattona ja selväkielisenä; tästä käy hyvänä esimerkkinä ftp-tiedonsiirrot sekä telnet-pääteistunnot. Tällaista liikennettä on helppo kuunnella jos hankkii fyysisen pääsyn sopivaan kohtaan verkkoa. Selväkielisyyden ongelma on suhteellisen helppo selättää hankkimalla esimerkiksi SSH-ohjelmiston, jolloin sekä tiedostonsiirrot että pääteistunnot ovat salattuja. TCP/IP-verkon heikkouksiin kuuluu käyttäjien todentaminen; perinteisesti todentaminen on tapahtunut salaamattomilla salasanoilla. Tällainen salaamattoman salasanan selville saaminen on helppoa. Ei ole olemassa murtovarmaa kassakaappia ja sama pätee tietoturvallisuuteen. Verkkoon tunkeutumisen ja yhteyksien kaappaamisen uhka on aina olemassa samoin kuin puhtaan kiusanteonkin, jossa esimerkiksi vain pyritään lamauttamaan palvelin. Tietoverkkohyökkäyksissä käytetään hyväksi protokollaan jääneitä aukkoja (Kaario 2002, 297). Mainitsen muutamia perinteisiä murtautumiskeinoja, jotka ovat aiheuttaneet harmia internetissä.

Palvelunestohyökkäyksessä (Denial of service) on tavoitteena palvelun tai palvelimen normaalin toiminnan estäminen, jolloin kyseessä onkin lähinnä kiusanteko. Kotikäyttäjien laimin lyömä tietoturvasta huolehtiminen on johtanut siihen, että monet kotikoneet ovat saastuneet viruksista, jotka tietyllä hetkellä osallistuvat tällaiseen palvelunestohyökkäykseen, jolloin kyseessä on hajautettu hyökkäys (Distributed denial of service, Ddos). Palvelunestohyökkäyksessä käytettyjä menetelmiä ovat esimerkiksi SYN Flood ja Ping of death. (Kaario 2002, 297.)

SYN-tulvassa (SYN Flood) palvelimelle lähetetään suuri määrä yhteydenmuodostuspyyntöjä ja vastauksiin jätetään vastaamatta. Tällöin palvelimelle syntyy suuri määrä puoliavoimia yhteyksiä. Jokainen puoliavoin yhteys syö palvelimen resursseja, jotka jossain vaiheessa loppuvat. Esimerkiksi hajautettua hyökkäystä on vaikea torjua ilman, että samalla estetään todellisten asiakkaiden yhteydenmuodostuspyyntöjä. (Kaario 2002, 298.)

Ping of death-hyökkäyksessä käytetään hyväksi ping-ohjelmaa, jolla kohdekoneelle lähetetään ylimittainen paketti. Standardi määrittelee paketin sisältämän datan määräksi 64 kilotavua. Kun määrä menee tämän yli voi pahimmassa tapauksessa sanoman vastaanottava laite kaatua, käynnistyä uudelleen tai vähintään menee muulla tavalla epämääräiseen tilaan. (Kaario 2002, 298.)

Muita tunnettuja palvelunestohyökkäykseen käytettyjä tekniikoita ovat mm. UDP echo, Smurf, Teardrop, Land, Latierra, Papasmurf, Targa, RST- ja FIN (secmeter.com). Näistä tekniikoista ja niiltä suojautumisesta löytää tietoa internetistä.

Kaikki osa-alueet vaikuttavat toisiinsa ja niillä on yhteisiä tekijöitä, joten jaottelu varmasti vaikuttaa kömpelölle ja on hyvin keinotekoinen. Tietoturvan suunnittelua ja määrittelyä kuitenkin auttaa asioiden jaottelu edes jollain tavalla. Suunnittelussa on havaittava riskit, joiden välttämiseen pyritään löytämään keinot. Suunnittelu ja tietoturvan toteuttaminen on tasapainottelua kustannustekijöiden, saavutetun hyödyn sekä käytettävyyden kanssa. Lisättäessä palvelutasoa, joustavuutta ja käytettävyyttä lasketaan samalla yleensä turvallisuustasoa (Hakala ym. 2006, 17).

3 TIETOTURVAN TOTEUTUS

Edellisessä luvussa käsittelin tietoturvallisuuden eri osa-alueita. Seuraavassa luvussa käsittelen eri asioita, mitä tulee ottaa huomioon tietoturvaa toteutettaessa ja käytän samaa jaottelua kuin Tietoturvallisuuden käsikirjassa. Osaltaan olen käsitellyt tietoturvan toteutusta edellisen luvun alakohdissa.

3.1 Työasematurvallisuus

Työasemien käytön turvallisuus on riippuvainen käyttäjän osaamisesta ja motivaatiosta (Hakala ym. 2006, 124). Olennainen osa työasematurvallisuutta ovat käyttöoikeudet ja niiden hallinta edellyttää käyttäjällä olevan henkilökohtainen tunnus työasemalle kirjautumista varten. Yleisimmin kirjautuminen on toteutettu käyttäjätunnuksella ja siihen liittyvällä salasanalla. Kirjautumisessa voidaan vaatia tunnuksen ja salasanan lisäksi älykortin (vrt. HST-kortti) käyttöä tai biometristä tunnistusta. Käyttöoikeuksiin liittyy myös niiden laajuus. Yleensä käyttäjille annetaan vain peruskäyttäjäoikeudet, joita laajennetaan tarpeen mukaan. Linux-käyttöjärjestelmässä tiedostojärjestelmien käyttöoikeudet esitetään maskina, jonka rakenne on: omistaja, ryhmä ja muut. Esimerkiksi vain pääkäyttäjälle muokattavissa oleva mutta kaikille luettavissa oleva tiedosto olisi listauksessa seuraavan näköinen:

```
-rw-r--r--  1 root          root  24072 2010-05-29 09:25 faillog
```

Maski näkyy rivin alussa, -rw-r–r--, jossa r tarkoittaa lukuoikeutta ja w luonnollisesti muokkausoikeutta. Tiedoston omistaja ja ryhmä ovat pääkäyttäjä, root. Linux-järjestelmissä ei ole erillistä järjestelmärekisteriä kuten Windowsissa, vaan järjestelmän ja ohjelmistojen asetukset sijaitsevat erillisissä tiedostoissa. Näin oikeuksien hallinta voidaan tehdä tiedostojärjestelmän kautta.

Työasematurvallisuuteen liittyy myös standardointi – vakioidut työasemat niin laitteiston kuin ohjelmistojenkin suhteen mahdollistavat helpomman työaseman palauttamisen ja monistamisen. Tiedon varmistusta helpottaa, jos työt tallennetaan organisaation ohjeistuksen mukaiseen paikkaan. Helpointa lienee luoda jokaiselle oma henkilökohtainen verkkolevy, josta otetaan varmuuskopio robotin toimesta määräajoin.

Käyttöjärjestelmien ja ohjelmistojen keskitettyihin päivityksiin ei Linux-puolella ole montaa ohjelmistoa tarjolla; yhtenä esimerkkinä toimii Redhat Network. Se on keski-

tetty hallintajärjestelmä RedHat Linux-järjestelmien käyttöön ja se vaatii lisensioidun Enterprise-version. Päivitysjärjestelmällä hallitaan ohjelmistojen päivityksiä, poistoa ja uusien ohjelmistojen asennuksia. Käyttö rajoittuu jakeluun kuuluviin ohjelmistoihin.

Virustorjuntaakaan ei sovi unohtaa. F-Secure tarjoaa nykyään myös linuxille turvaratkaisuja esimerkiksi F-Secure Linux Security-paketin, jossa on reaaliaikainen virus- ja haittaohjelmatarkistus, palomuuri sekä tarkkailuprosessit tunkeutumisen havaitsemiseksi. Lisäksi pakettiin kuuluu keskitetty hallintajärjestelmä F-Securen tietoturvaohjelmistoille. Ohjelmistossa on raportointijärjestelmä, joka lähettää hälytyksen hallintapalvelimelle kaikesta havaitusta normaalista poikkeavasta tilanteesta
(HIPS, www.f-secure.com 13.11.2011,
http://www.f-secure.com/fi/web/business_fi/products/servers/solution).

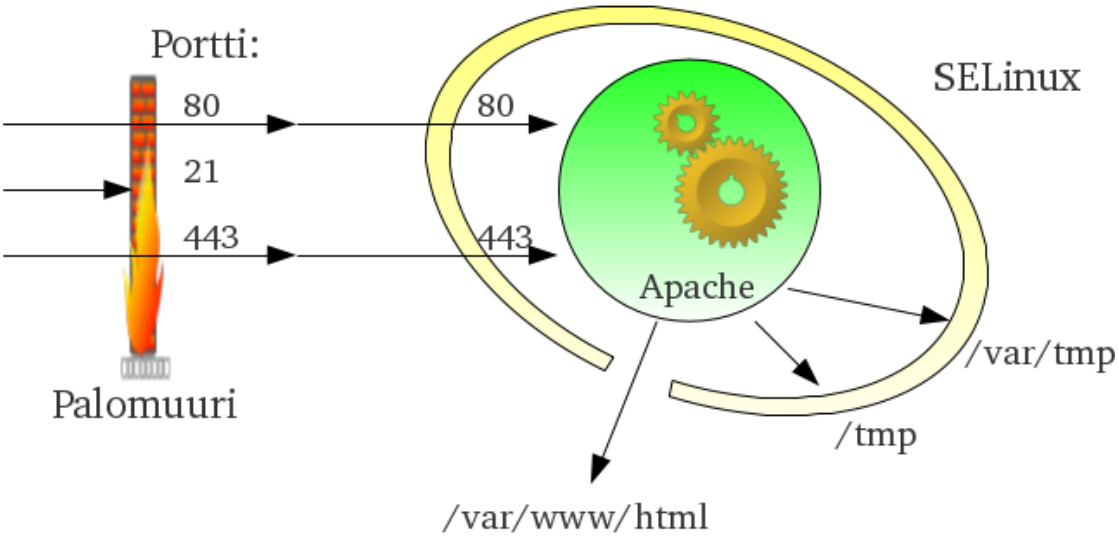
3.2 Palvelinturvallisuus

Huolimatta siitä miten yritys on valinnut palvelinympäristönsä, on tietyt perusasiat ratkaistava ja niiden myötä syntyneet tarpeet täytettävä. Laitteiston on oltava vikasietoinen; palvelut eivät saa keskeytyä yksittäiseen laiterikkoon. Vikasietoisia laitteistoratkaisuja ovat esimerkiksi kahdennetut virtalähteet, muistipiirit, tuulettimet sekä RAID-levyjärjestelmät. Muistin osalta kannattanee suosia virheitä korjaavaa muistia, joka tarkistaa muistiin tallennetun tiedon eheyden ja korjaa pienimmätkin yhden bitin virheet.

Varmistaminen kuuluu myös oleellisesti perustoimintoihin. Varmistamisen toteutus vaatii oman huolellisen suunnitelmansa aina varmistustietojen säilytysratkaisuihin asti siten, että ne ovat turvassa ulkoisilta uhilta.

Palvelinturvallisuuteen kuuluu myös valvonta, auditointi, jossa seurataan käyttöjärjestelmän toimintaa ja erilaisten käyttöoikeuksien käyttöä (Hakala ym. 2006, 156). Linux-järjestelmissä tapahtumavalvonta löytyy järjestelmän lokitiedoista. Oletuksena järjestelmän keräämät tiedot eivät ole kovin laajat. Tiedostojärjestelmän käytön seurantaan voi käyttää apuna Tripwire-järjestelmää (Wikipedia, Open Source Tripwire). Kyseisellä ohjelmalla jäljitetään tiedostoissa ja hakemistoissa tapahtuneita muutoksia. Linux-järjestelmän turvallisuutta voidaan parantaa monella tapaa, kuten esimerkiksi edellisessä luvussa mainitulla F-Securen ratkaisulla tai jollain muulla ohjelmistolla, joita on useita vaihtoehtoja; AppArmor, Bastille Linux, Tomoyo. Yksi useimmin käytetyistä on alunperin Yhdysvaltain NSA:n projektiin pohjautuva SELinux, Security-Enhanced Linux, jolla määritellään policy-asetuksia, jotka koskevat tiedostojärjestelmää sekä

järjestelmässä toimivia ohjelmistoja, palveluja ja käyttäjiä (Wikipedia, SELinux). Sillä rajoitetaan kunkin kohteen oikeuksia, jotta vaikutukset jäisivät mahdollisimman vähäisiksi virheen tai väärän asetuksen vuoksi. Jokainen tiedosto, prosessi, hakemisto ja portti saa oman kontekstin, jonka mukaan sille myönnetään oikeuksia. Käytettäessä Apache-ohjelmaa http-palvelun tuottamiseen, voi ilkeämielinen käyttäjä käyttää hyväksi ohjelmistossa mahdollisesti olevaa heikkoutta ja saa käyttöönsä ohjelman oikeudet – tässä tapauksessa käyttäjän ja ryhmän apache oikeudet. SELinuxin säännöillä voidaan estää apache-käyttäjän pääsy muualle kuin /var/www/html-hakemistoon, jossa normaalisti pidetään www-sivustojen tiedostot (kuva 2). Palomuurilla estetään ulkopuolisten pääsy koneelle muuten kuin hyväksytyistä porteista ja SELinuxilla estetään palvelimella tai työasemalla ajettavien sovellusten pääsyä portteihin, tiedostoihin ja hakemistoihin. SELinuxia voidaan käyttää myös MLS-asetuksilla, Multi Level Security, jolloin sillä rakennetaan koneen sisäiset tietoturvaluokat. Tällä voidaan tyystin rajata sovelluksia omaan ympäristöön ja jopa omaan muistiavaruuteen.



KUVIO 2. Periaatteellinen kuva SELinuxin toiminnasta.

Edellisessä luvussa tuli esille keskitetyt päivitysratkaisut virustorjunnan ja ohjelmistopäivitysten osalta. Järjestelmäasetuksien tarkistukseenkin löytyy apuohjelmia kuten Nessus. Se on tarkoitettu verkossa olevien palvelimien ja palvelujen turvallisuuden testaamiseen. Se suorittaa määriteltyjä testejä turva-aukkoihin liittyvien tietojen pohjalta, jotka se noutaa päivitetystä tietokannasta. (www.nessus.org.)

3.3 Verkon turvallisuus

Verkon turvallisuutta lähdetään rakentamaan aluksi mekaanisesti. Suojautumiseen kuuluu palvelinhuoneiden lukitseminen, eristäminen ja kaapelien sekä liittimien suojaaminen asiattomilta ja valvomattomalta käytöltä. Verkon rakenne on päätettävä ennen kuin verkon laitteistollisia ja ohjelmallisia suojaamistoimia voidaan suunnitella. Toiminnan kannalta on tärkeää, ettei synny liikenteellisiä pullonkauloja ja ettei sisäverkkoon pääse ulkoa käsin muuten kuin suojattuja yhteyksiä pitkin.

Nykyisin on yleisesti käytössä toteutustapa, jossa verkkoon muodostetaan ns. demilitarisoitu alue (DMZ) eli eteisverkko. Tällä tarkoitetaan yleensä asiakkaille suunnattua julkista extranet-aluetta. Eteisverkon käyttö on valtionhallinnon verkkoarkkitehtuurissa pakollista (VAHTI 3/2010). Tähän verkon alueeseen liitetyt koneet voivat joutua hyökkäyksen kohteeksi ilman että se välttämättä lamauttaisi organisaation toiminnan. Internet-palveluiden kuuluessa yrityksen toimenkuvaan tulee palveluiden käytettävyyden takaaminen olemaan haasteellinen tehtävä tässäkin toteutustavassa. Usein DMZ-alueeseen sijoitetaan WWW-palvelin, ensimmäinen SMTP-postipalvelin sekä DNS-palvelin. Tällöin DMZ-alueen postipalvelin ottaa vastaa sähköpostia ja lähettää sen edelleen sisäverkon postipalvelimelle. Postipalvelimien konfigurointiin on paneuduttava huolella. Postipalvelimesta on muistettava poistaa reititys – tällöin poistetaan virheellisten osoitetietojen sisältämien sähköpostien aiheuttama viestitulva. Samoin suositellaan jaetun postin tietokannan käyttöä, jolloin useammalle henkilölle lähetetystä viestistä tallennetaan vain yksi kopio.

Eteisverkon DNS-palvelin huolehtii ainoastaan eteisverkkoon kuuluvien laitteiden DNS-nimistä. Siinä ei saa olla sisäverkon koneiden tietoja. DNS-palvelimet käyttävät UDP-protokollaa, josta pakettisuodatuksella toteutettu palomuuri ei pysty päättelemään yhteyden muodostussuuntaa. Tämän vuoksi käytetään kahta DNS-palvelinta, jottei sisäverkon UDP-porttia tarvitse jättää auki - DNS-palvelu käyttää ainoana sisäverkon palveluista UDP-protokollaa. DMZ-alueen DNS-palvelin suorittaa sisäverkon ulkopuolisia osoitteita koskevat kyselyt. Sisäverkon DNS-palvelin toimii orjapalvelimena ohjaten kaikki sisäverkon ulkopuoliset kyselyt julkiselle DNS-palvelimelle. Ainoastaan julkisen DNS-palvelimen sallitaan liikennöidä sisäverkon DNS-palvelimen UDP-porttiin – kaikki muu UDP-liikenne kielletään. Kaikki internetistä tulevat sisä- tai eteisverkon osoitteita lähdeosoitteinaan käyttävät paketit suodatetaan myös pois (IP spoofing).

Jokaiseen verkkorakenteeseen kuuluu yleensä palomuurit ja suosittelen sitä myös kotona sijaitsevaan työasemaankin. Palomuurit toimivat sisäverkon ja ulkoverkon välisenä rajana estäen liikennöinnin sisäverkkoon ilman asiaankuuluvia oikeuksia. Toiminnallisesti palomuurit voidaan jakaa kolmeen perustyyppiin: pakettisuodattimiin, välityspalvelimiin ja sovellustason yhdyskäytäviin (Hakala ym. 2006, 187). Pakettisuodatin toimii TCP/IP-protokollapinon verkkokerroksella hyläten liikennettä lähde- ja kohdeosoitteiden sekä sovellusten käyttämien porttinumeroiden perusteella. Pakettisuodatinta käytetään yleisesti reitittimissä, joihin on helppo konfiguroida pääsylistoja – ei täydellinen turvaratkaisu, mutta täydentää muilla protokollakerroksilla toimivia suotimia.

Välitys- tai ns. proxypalvelimet toimivat protokollapinon kuljetuskerroksella avaten käyttäjän puolesta yhteyden johonkin palveluun. Se tunnistaa UDP- ja TCP-protokollan porttinumerot ja tekee niiden ja ennalta konfiguroitujen tietojen perusteella päätöksen mahdollisesta liikenteen hylkäämisestä.

Tehokkaimpia palomuuriratkaisuja ovat sovellustason yhdyskäytävät. Se käyttää suodatuspäätöksissään kaikkien kerroksien tietoa aina sovelluskerroksella saakka (Kaario 2002, 306). Sovellustason yhdyskäytävä tutkii jokaisen paketin sisällön välittäessään liikennettä asiakas- ja palvelinohjelmiston välillä. Tämän vuoksi laitteelta edellytetään huomattavaa prosessoritehoa.

Kaikkein uusimmat palomuurit tallentavat tietoa yhteyden osapuolten välisestä sanoman vaihdosta. Tällöin puhutaan tilallisesta suodatuksesta, joka avaa mahdollisuuksia erilaisille suodatusmenetelmille ja uusille sovelluksille palomuurin rinnalle. Esimerkkinä voisi toimia laskutuksen toteuttaminen palomuurin keräämien liikennetietojen avulla (Kaario 2002, 307).

Kaikki palomuuritekniikat edellyttävät pakettisuodatusmenetelmien tuntemista. Pakettisuodatuksen perustietoja; lähde-, kohdeosoitteita ja porttinumeroita käyttämällä määritellään hyväksyttävä liikenne. Linuxissa on useita ohjelmistoja käytettävissä pakettisuodatukseen ja uusimpana on käytössä iptables. Iptables kykenee tekemään osoitteenkäännöstä (NAT). Iptablesin toiminta voidaan jakaa liikenteen suodatukseen ja pakettien muokkaamiseen, joista jälkimmäistä tarvitaan osoitteenkäännöksessä (Hakala ym. 2006, 205). Palomuuriratkaisuissa on suositeltavaa kieltää ensin kaikki liikenne ja tämän jälkeen ruveta sallimaan tarpeellista liikennettä.

Verkon monitorointia käytetään tietoturvahyökkäyksien havaitsemiseen. Tällaiseen toimintaan on kehitetty Network Intrusion Detection System (NIDS), joka havaitessaan epänormaalia toimintaa konfiguroi esimerkiksi reitittimen tai palomuurin niin, että epänormaali toiminta käy mahdottomaksi. Vapaan ohjelmakoodinkin NIDS-ohjelmia on saatavilla, kuten Snort (http://fi.wikipedia.org/wiki/SNORT).

Vahva salaus on Internetin tietoturvan perusta (Kaario 2002, 310). Vahvan salauksen menetelmiä ovat mm. virtuaaliset erillisverkot (VPN), WWW:n yhteydessä SSL ja SSH. Virtuaalisia erillisverkkoja käytetään, kun halutaan liittää turvallisesti toisiinsa yksityisiä lähiverkkoja ja mobiililaitteita julkisen verkon kautta vaikkapa etätyöskentelyssä. VPN-yhteys voidaan edullisesti toteuttaa esimerkiksi käyttäen päätepisteinä Linux-koneita tunneloiden näiden välinen liikenne SSH-yhteydelle PPP-protokollaa käyttäen. VPN-ratkaisuissa tulee muistaa palomuurin tarve vaikka yhteys onkin salattu. Yhteys olisi ilman palomuuria avoin. Varsinaiset salausalgoritmit jaetaan symmetrisiin ja epäsymmetrisiin menetelmiin. Käyttäjien tunnistus on tietoturvan kannalta käyttäjälle näkyvin osa. Salasanojen murtamista vastaan parhaimman suojan antaa vahvan salauksen kanssa riittävän usein vaihtuva ja vaikeasti arvattava salasana. Todellinen käyttäjän tunnistaminen tapahtuu esimerkiksi PKI-menetelmällä.

Olen näissä kahdessa luvussa tuonut esille vain pienen osan uhkakuvista ja erilaisista suojautumismekanismeista. Kaikkein tehokkain tapa lisätä tietoturvaa on kuitenkin käyttäjien kouluttaminen ja heidän asenteidensa kehittäminen. Pieni terve epäluulo kaikkea epänormaalia kohtaan lisää tietoturvaa paljon. Sosiaalinen hakkerointi luo tällä hetkellä valitettavasti parhaimmat tulokset – siihen ei mikään tekninen yksityiskohta auta.

3.4 Ympäristöturvallisuus

Ympäristöturvallisuutta eli ns. perinteistä turvallisuutta käsitellään ISO 17799 -standardin klausuulissa 9 (ISO 17799:2005). Ympäristöturvallisuus on tässä standardissa jaettu kahteen osaan: tila- ja laitteistoturvallisuuteen. Tilaturvallisuuden osalta käsitellään muun muassa:

- fyysisiä turvatoimia
- kulunvalvontaa
- lukituksia
- hälytysjärjestelmiä
- suojautumista ulkoisilta uhilta

- tulipaloja
- tulvimista
- ilmastointia (jäähdytys)

Nykyään konesalit tuottavat huomattavan määrän lämpöä, joten jäähdytykseen on kiinnitettävä erityistä huomiota. Tästä on viime ajoilta parikin esimerkkiä miten asia on huomioitu. Savon Voiman yritystalossa Siilinjärven Toivalassa ohjataan Enfon konesalin liikalämpö kesäaikaan maaperään porattuihin kaivoihin, joista taasen talvella kerätään lämpöä kiinteistön lämmitykseen. Lehtiartikkeleissa puhuttiin maalämmön sijasta maaviileästä. Toisena esimerkkinä on Googlen päätös rakentaa yksi konesali Suomeen vanhaan paperitehtaaseen, jossa etuna on tehtaan sijainti ilmaisen veden lähellä, jota voi käyttää jäähdytysjärjestelmissä.

Laitteistoturvallisuuden osalta standardi käsittelee esimerkiksi laitteiden sijoittelun avulla suojaamista ympäristöuhkia ja luvatonta käyttöä vastaan, ukkosilta suojautumista ja sähkön syötön turvaamista UPS-laitteilla. Standardissa muistetaan myös huomioida laitteistojen käytöstä poistamisen yhteydessä luottamuksellista dataa sisältävien medioiden poistaminen tai luotettava ylikirjoittaminen. Tästäkin olemme saaneet esimerkkejä, kun muun muassa potilastietoja on löytynyt käytöstä poistetuista ja kierrätykseen menneistä tietokoneista.

3.5 Sovellusturvallisuus

Sovellusturvallisuus, väheksymättä yhtään muita tietoturvallisuuteen liittyviä osa-alueita, on niin laaja käsite, että huomioin sen tässä työssä vain pintapuolisesti. Tietoturvallisuuden kolmen perustekijän on oltava jatkuvasti sovellussuunnittelijan ja -ohjelmoijan mielessä. Nämä perustekijät ovat tutuksi tulleet eheys, käytettävyys ja luottamuksellisuus. Perustekijöille on monenlaisia uhkia kuten inhimilliset virheet tietojen syötössä, virheelliset tulkinnat ja laitteiden aiheuttamat tekniset virheet sekä ohjelmavirheet. Luottamuksellisuus voi vaarantua aina tietoja tulostettaessa, siirrettäessä ja tallennettaessa. (Hakala ym. 2006, 319).

Sovelluskehityksen tietoturvallisuutta ohjaa standardi ISO 17799. Mahdollisesti suurin osa tapahtuneista virheistä ja tietojärjestelmissä olevista virheellisistä tiedoista on meistä käyttäjistä lähtöisin. Standardi tarjoaa kontrolleja, joilla näitä inhimillisiä virheitä voitaisiin vähentää. Esimerkiksi syötteen käsittelyyn voidaan lisätä erilaisia tarkistusmenetelmiä. Numeeriset tietojen suhteen kannattaa tarkistaa, onko käyttäjän an-

tama syöte luku. Lisäksi usein syötteelle määritetään minimi- ja maksimiarvo.(Hakala ym. 2006, 319-322.)

Käyttäjälle voidaan tarjota vaihtoehtoja eli rajataan mahdollisten annettavien arvojen lukumäärää, joista hän valitsee halutun. Edellisellä tavalla myös estetään väärinkirjoittamisen mahdollisuus. Tarkistusmerkkejä voidaan myös käyttää, jolloin vertaillaan syötteeseen sisältyvää tarkistetta ohjelmallisesti laskettuun tarkisteeseen. Voidaan esimerkiksi laskea tarkiste henkilötunnuksesta ja tarkistaa, että käyttäjän antama syöte kuuluu käsiteltävien tietojen luokkaan – tässä tapauksessa vakiomuotoisiin henkilötunnuksiin. Käyttäjän syötettä voidaan rajata myös peitteellä, esimerkiksi haettaessa käyttäjältä hänen osoitteensa postinumeroa, on syötteen oltava muotoa 00000 (nollan tarkoittaessa numeroa). (Hakala ym. 2006, 319-323.)

Käyttäjä tekee helposti virheitä tulkitessaan ohjelman antamia tulosteita. Näitä virheitä voidaan estää huolellisella käyttöliittymäsuunnittelulla. Käyttöliittymän tulisi olla selkeä ja yksinkertainen, josta tarvittava tieto tai asia löytyy helposti ilman tulkinnanvaraisuutta. Sama pätee paperitulosteisiin, joita otetaan toistuvasti rutiinitoimintoihin. Tarvittaessa voidaan tietojen käsittelyssä käyttää viivakoodeja. (Hakala ym. 2006, 324.)

Tietoja siirrettäessä ja tallennettaessa voivat tiedot muuttua tai niiden sisäinen järjestys voi muuttua siirrettäessä tietoa pieninä sanomina reitittimien välittämänä. Virheiden havaitsemiseen on useita tekniikoita – voidaan käyttää tiedoista laskettavaa yksinkertaista tarkistetta kuten tarkistussumma tai korjausmahdollisuuden antavaa tarkistetta. Eheyden tarkistamiseen voidaan käyttää tiivisteitä. Monet tiivisteet ja yksinkertaiset tarkisteet mahdollistavat virheen havaitsemisen, mutta eivät tarjoa keinoja virheen korjaamiseen automaattisesti. Yksittäisen bitin tai merkin korjaamiseen voidaan käyttää laskenta- ja palautusalgoritmeja hyödyntäviä tarkisteita ja tiivisteitä. Varsinaisia korjausalgoritmeja hyödyntävät tarkisteet ja tiivisteet vaativat paljon laskentatehoa. Tietoja kopioitaessa on syytä muistaa aina tarkistaa, että alkuperäinen ja kopioitu tiedosto ovat identtiset. Kaikki kopiointitavat eivät suorita tarkistusta. Salausta voidaan käyttää estämään tietojen luvatonta käyttöä. (Hakala ym. 2006, 325-330.)

Käytettävyys sovelluksissa on käsitettävä tietojen saatavuutena ja käyttökelpoisuutena. Se pakottaa pohtimaan ohjelmointiteknisiin asioihin sekä käyttöliittymään ja tietojen esitysmuotoon liittyviä ratkaisuja. Laitteistojen ja verkon kuormitukseen sekä sovelluksen nopeuteen on kiinnitettävä huomiota. Lisäksi käyttäjien on erikseen suori-

tettava sovellukselle käytettävyystestaus teknisen testauksen lisäksi. (Hakala ym. 2006, 336-337.)

Tietokantaohjelmointia ohjaavat myös tietoturvallisuuden kolme perustekijää. Yleisimmin suurissa tietojärjestelmissä käytetään relaatiotietokantoja. Tietojen eheys ja käytettävyys varmistetaan niissä omilla mekanismeilla. Luottamuksellisuus otetaan huomioon käyttöoikeuksien suunnittelussa ja jakamisessa.

Tietoturvallisuuden huomioimisessa tietokannoissa on avainasemassa tietokannan käsitemalli eheyssääntöineen. Käsitemalli eheyssääntöineen on yksi tietokannan keskeisimpiä määrityksiä antaessaan puitteet muulle eheysmäärittelylle (Hakala ym. 2006, 340). Tietokannoissa eheyttä ylläpidetään viite-eheysmäärityksillä ja -säännöillä. Viite-eheys estää viittaukset tietoihin, joita ei ennestään ole olemassa. Kustannustehokkainta on toteuttaa eheysmääritykset tietokantatasolla, mutta mikään ei estä ylläpitämästä viite-eheyttä ohjelmakoodissa. Viite-eheys toteutetaan luomalla taulujen välille liitokset, joilla kerrotaan millainen yhteys on kahden eri taulun välillä. Liitosten käyttäytymistä voidaan ohjata lisämääreillä kuten johdannaispoistolla. Johdannaispoistossa käyttäjän poistaessa perusavaimen päätaulusta siihen liitetystä taulusta poistetaan kaikki perusavainta viiteavaimena käyttäneet tietueet. (Hakala ym. 2006, 338-344.)

Eheyden ylläpitoon käytetään myös tietokannan hallintajärjestelmän tarjoamia tarkistus- ja virheilmoituspalveluita. Transaktiolokeja käytetään tapahtumatietojen palauttamiseksi sähkökatkojen tai järjestelmän virhetilanteiden jälkeen. Taulujen tietoja voidaan tarkistaa käyttämällä syötteen rajoittamista ja tarkistusmekanismeja. (Hakala ym. 2006, 343-344.)

Tietokannan rakenteella on olennainen merkitys tietojärjestelmän nopeudelle. Indeksoinnilla voidaan nopeuttaa usein suoritettuja kyselyitä. Kyselyiden suorituskykyä voidaan mitata erilaisilla profilointityökaluilla kuten Microsoftin SQL Serverin yhteydessä SQL Query Analyzer ja SQL Profiler. Profilointityökaluilla saadaan tietoa suorituskyvystä ja voidaan havainnoida edellä mainitun indeksoinnin vaikutusta tietojen hakunopeuteen. (Hakala ym. 2006, 345.)

Valittaessa sovellukselle tietokantaa on syytä huomioida siirrettävän tiedon määrä. Esimerkiksi Access-sovelluksessa tietokantapalvelimen tiedot linkitetään sovellukseen, jolloin sovelluksen kannalta taulut toimivat kuten ne olisi tallennettu työaseman kiintolevylle. Tällöin tietojen lukeminen saattaa kuormittaa verkkoa huomattavan pal-

jon tietokannan koon kasvaessa. Peruslähtökohdaksi onkin otettava kyselyiden suorittaminen tietokantapalvelimella ja vain tulosjoukon lähettäminen verkon yli. Access-tietokantasovelluksissa voidaan käyttää läpivientikyselyitä suuriin tauluihin kohdistuvissa kyselyissä, jolloin saadaan verkon kuormitusta vähennettyä. (Hakala ym. 2006, 346-347.)

Tietokannan käyttöoikeusmäärityksillä ylläpidetään luottamuksellisuutta. Oikeuksien määrittämiseen voidaan käyttää käyttäjätietokantaa tai hakemistopalvelun käyttäjätietoja, jos sellainen on käytössä. Tietokantatasolla voidaan käyttöoikeuksia määritellä yksittäiseen taulun sarakkeeseen asti. (Hakala ym. 2006, 348-349.)

4 LAMP

Sivuston voi toteuttaa monellakin eri ohjelmistolla, mutta täyttääkseni edullisuuden vaatimuksen, valitsin itse alustaratkaisuksi LAMP-kokonaisuuden. Osin valintaan vaikutti myös omat mieltymykseni sekä aiempi Linuxin käyttö. LAMP on palvelinympäristö, joka muodostuu neljästä eri ohjelmistosta. Käyttöjärjestelmänä on linux, Apache toimii HTTP-palvelimena mahdollistaen WWW-sivujen näyttämisen WWW-selaimen kautta, tietokantana on MySQL ja skriptauskielenä PHP, jolla on mahdollista luoda monipuolisia dynaamisia sivustoja. LAMP tarjoaa monia etuja ja mahdollisuuksia, kun sitä vertaa moniin muihin, esimerkiksi Microsoftin, tarjoamiin ratkaisuihin. Kaikki ohjelmistot ovat saatavilla maksutta lähdekoodin kanssa. Kyseessä on hyvin joustava toimintaympäristö ja se onkin muodostunut erittäin suosituksi; esimerkkinä ratkaisun käyttäjästä toimii Wikipedia. Jo yhden palvelimen LAMP-ratkaisulla voidaan palvella tuhansia käyttäjiä samanaikaisesti ja muodostamalla palvelinryppäitä voidaan puhua kymmenistä tai sadoista tuhansista käyttäjistä.

Valitsemani testiympäristö muodostuu eteisverkon palvelimesta, joka toimii palomuurina ohjaten käyttäjän HTTP-pyyntöjä demilitarisoidun alueen www-palvelimelle sekä sisäverkon tietokantapalvelimelle. Käyttöjärjestelmänä toimii Linux, www-palvelinohjelmana Apache, tietokantana MySQL ja skriptikielenä PHP. Näin toimien hallitaan tietoturvaa paremmin ja parannetaan sivuston nopeutta.

4.1 Linux

Tarkasti ottaen Linuxilla tarkoitetaan vain itse kerneliä ja GNU/Linuxilla tarkoitetaan koko käyttöjärjestelmää komentotulkista ikkunointijärjestelmään. Linux-ydin julkaistaan hyvin avoimen GPL-lisenssin alla. Se antaa oikeuden käyttää ohjelmaa kuten haluaa, tutkia ja muuttaa ohjelmaa sekä parannella ohjelmaa, jolloin parannukset on julkaistava kaikkien saataville.

Itse valitsin testiympäristööni CentOS-nimisen jakeluversion ilman graafista käyttöliittymää. Tämä johtui pitkälti siitä, että ajoin palvelimia virtuaaliympäristössä, joten testipalvelimet oli toteutettava kevyesti. Tietoturvassa usein vähemmän on enemmän eli mitä vähemmän on palveluita käynnissä, sitä vähemmän on myös mahdollisia turva-aukkoja. Tämän vuoksi poistin palvelimista kaikki turhat palvelut käytöstä (netstat -tuanp).

Linux-palvelimia varten on eri julkaisuja ja apuohjelmia, joilla niiden turvallisuutta voidaan parantaa ohjeistetusti. SELinux on asennuspaketti sisältäen käyttäjälle suunnattuja apuohjelmia ja modifioidun linuxin ytimen. Se käyttää hyväksi linuxin ytimen turvallisuusmoduleita (LSM, Linux Security Modules). SELinux tarjoaa tuen käyttäjien ja järjestelmän oikeuksien normaalia tarkempaan hallintaan. Bastille Linux ja sittemmin Bastille Unix on komentoikkunassa suoritettava käyttäjää opastava skripti, joka paikkaa mahdollisia tietoturva-aukkoja käyttöjärjestelmästä. Ilmeisesti Bastillea ei enää kehitetä (Wikipedia, Bastille Unix).

4.2 Apache

Apache on erittäin suosittu, joustava ja tehokas viimeisimpiä protokollia käyttävä palvelinohjelmisto. Se on myös erittäin muunneltava ja laajennettavissa oleva. Itse asiassa Apache on Internetin suosituin HTTP-palvelinohjelmisto jo vuodesta 1996 lähtien (Wikipedia, Apache). Apache julkaistaan omalla lisenssillään, joka on myös hyvin avoin – lähdekoodin saa ladata ilmaiseksi ja sen käyttö on ilmaista. Lähdekoodia saa muokata, tehtyjä muutoksia ei tarvitse julkaista ja niitä saa myydä eteenpäin.

4.3 MySQL

MySQL on avoimen lähdekoodin tietokantaohjelmisto. Se julkaistiin vuonna 1995 ruotsalaisen yhtiön toimesta. Nyttemmin omistus on siirtynyt Oraclelle. MySQL on suosittu WWW-sivujen tietokantana helppoutensa ja edullisuutensa vuoksi. MySQL

julkaistaan GPL-lisenssin alla, joten lähdekoodi on vapaasti saatavilla. Ilmaisen lisenssin mukana ei tule tukea.

4.4 PHP

PHP, Hypertext Preprocessor, on yleiskäyttöön tarkoitettu ja hyvin suosittu avoimen lähdekoodin skriptauskieli, joka voidaan upottaa HTML-sivujen sisälle. PHP:n lisenssi myöntää oikeuden jaella lähdekoodia ja binäärejä alkuperäisen lisenssin mukana. Lisenssi kieltää PHP-nimen käytön muunneltujen ohjelmistojen yhteydessä ilman eri lupaa.

PHP konfiguroidaan tiedostossa php.ini, josta tulee suurin osa sen tietoturvasta. Konfigurointitiedostolla voidaan kuitenkin lähinnä rajoittaa sitä, mitä PHP-skripteillä voidaan tehdä. PHP:n ytimeen kohdistetuille hyökkäyksille ja SQL-injektioille ei voida mitään php.ini-tiedostolla. Tämän vuoksi on perustettu projekteja, joilla PHP:n tietoturvaa kyettäisiin parantamaan. Joka tapauksessa suurin vastuu turvallisuudesta lankeaa sivuston koodaajalle ja suunnittelijalle. On tärkeää, että kaikki tietokantaan menevät arvot kuten käyttäjän sivulla antamat syötteet kulkevat jonkinlaisen tarkistuksen läpi (Heinisuo & Rauta 2007, 400). Tähän voidaan käyttää esimerkiksi mysql_real_escape_string-funktiota ja lisäksi kaikki syötteet on suljettava heittomerkkeihin sql-hauissa. Näiden kahden keinon avulla voidaan jo torjua monta virhetilannetta, tahallista tai tahatonta.

Jos käyttäjän syöttämien arvojen tulee olla kokonaislukuja, voidaan syötteet tarkastaa esimerkiksi PHP:n intval-funktiolla, joka palauttaa sille annetun parametrin kokonaislukuna. Tekstisyötteen muuttaminen ei onnistu, ja funktion tulos saa arvon nolla. Toinen funktio, ctype_digit, puolestaan soveltuu numeerisuuden tarkistamiseen – merkkijono ”222” palauttaa toden, kun taas esimerkiksi ”alfa3” tai ”3.1415” palauttavat epätoden.

Mysql_real_escape_string-funktiota on käytettävä sql-lausekkeeseen lisättävien syötteiden siistimiseen MySQL-rajapinnan yhteydessä. Tälle rajapinnalle on vaihtoehto, PDO-lisäosa - PDO tulee sanoista PHP Data Objects. PDO on tietokantariippumaton tapa käyttää tietokantaa (Heinisuo & Rauta 2007, 274). Se helpottaa eri tietokantojen käyttämistä, koska monet perustoiminnot ovat samanlaisia eri tietokannoissa. Jotkin ominaisuudet kuten MySQL:n limit-määre eivät toimi muissa tietokannoissa. PDO tuo mukanaan monia etuja kuten valmisteltujen lausekkeiden käytön – MySQL-rajapinnan kanssa tietokantahaut on kirjoitettava kokonaisuudessaan. Valmisteltuja

lausekkeita voidaan käyttää eri yhteyksissä uudelleen. Lisäksi niiden kanssa lausekkeisiin sijoitettavien arvojen siistiminen voidaan jättää tekemättä mysql_real_escape_string-funktiolla. PDO tekee käyttäjän syöttämän tiedon vaarattomaksi lisäämällä tarvittavat erikoismerkit.

5 SIVUSTON SUUNNITTELU

Työn perusvaatimuksina oli toteuttaa helppokäyttöinen sovellus WWW-ympäristöön yhdistyksen tietojen päivittämiseen ja arkistointiin. Perustoimintoja tulisi olemaan yleistietojen, kuten yhdistyksen, hallituksen kokoonpanon ja tapahtumien esittely, jäsentietojen ylläpito, kokouksien pöytäkirjojen arkistointi ja yhdistyksen tuotteiden tilaaminen. Palvelinympäristön ja käytettävien ohjelmistojen tulisi olla edullisia tai jopa ilmaisia. Tämän ja oman kiinnostukseni vuoksi valitsin ilman eri vertailuja toteutuskokoonpanoksi LAMP-ratkaisun, jota on edellisissä luvuissa sivuttu. Mikäli yhdistyksellä ei ole mahdollisuutta rakentaa omaa palvelinympäristöä tai jäsenien työnantaja ei anna mahdollisuutta sen rakentamiseen työnantajan tiloihin on LAMP-ratkaisut kaupallisilta tarjoajiltakin suhteellisen edullinen ratkaisu.

Aloitin suunnittelun olemassa olevien tarpeiden pohjalta kuvitellen mahdollisia käyttötapauksia, joita jäsenille tai ylläpitäjille voi tulla vastaan. Käyttötapauskaavioiden ja -kuvausten perusteella oli helppoa lähteä suunnittelemaan toimintoja ja tarvittavia tietokantarakenteita.

5.1 Käyttötapaukset

Liitteessä 1 kuvataan normaalin sivuston käyttäjän; satunnaisen kävijän tai yhdistyksen jäsenen, käyttötapaukset. Käyttötapauksia ovat jäseneksi hakeminen ja jäsenille lisäksi sisään kirjautuminen. Sisään kirjautumisen jälkeen mahdollisia käyttötapauksia olisivat kokoukseen ilmoittautuminen, merkkien ja muiden tuotteiden tilaaminen sekä omien tietojen hallinta.

Pääkäyttäjän tai ylläpitäjien kaaviossa (liite 2) on tapauksina jäsentietojen ja hallitustietojen ylläpitäminen sekä jäsenten tapahtumiin osallistumisaktiivisuuden, tilausten ja jäsenhakemusten seuraaminen. Näiden jatkoksi on mentävä syvemmälle; esimerkiksi jäsentietojen ylläpitäminen pitää sisällään uuden jäsenen tietojen lisäämisen ja vanhan jäsenen tietojen muokkaamisen. Käyttötapausten perusteella voi ryhtyä helposti pohtimaan eri toimintoja ja tarvittavia tietorakenteita.

5.2 Tietokanta

Aloitin tietokannan suunnittelun pohtimalla, mitä tietoja yhdistyksen tarvitsee ylläpitää rekisterissä sekä mitä tietoja tarvittaisiin kunkin käyttötapauksen toimintojen toteuttamiseen. Apuna olisin voinut käyttää olemassa olevaa jäsenrekisteriä, joka on Microsoftin Access-tietokannassa. Kyseisessä tietokannassa oli useita kenttiä, joiden tarkoitus on jo painunut unholaan. Tämänkin vuoksi on syytä pohtia tietokannan rakennetta uudestaan.

Tutustuttuani tässä vaiheessa paremmin yhdistyksen käytössä olevaan kirjanpito- ja tietokantaohjelmaan tulimme yhdessä yhdistyksen sihteerin kanssa siihen tulokseen, että ei ole tarkoituksenmukaista korvata käytössä olevaa ohjelmaa vaan luoda pelkistetty web-pohjainen sovellus yhteystietojen tarkastamiseen ja tiedon jakamiseen. Yhdistyksen käytössä oleva vanha, kaupallinen sovellus on riittävän hyvä ja käyttökelpoinen jäsentietokannan ja laskutuksen ylläpitoon. Samalla tietoturvan kannalta arimmat tiedot ovat paremmassa tallessa sihteerin vastuulla olevalla kannettavalla tietokoneella ja paperisessa kirjanpidossa. Mahdollisesta internettiin kytkettävästä tietokantasovelluksesta tulisi täten vain apuväline yhteystietojen tarkastamiseen, tilauskanava mahdollisten myytävien tuotteiden ostamiseen ja tietokanava hallitukselta jäsenien suuntaan. Muita mahdollisia toimintoja olisivat ainakin jäseneksi hakeminen ja kokouksiin ilmoittautuminen sekä kokous- ja hallitustietojen selaaminen. Mikäli halutaan seurata jäsenen aktiivisuutta osallistua yhdistyksen kokouksiin, tulee meidän luoda taulu, joka yhdistää jäsenen tiedot kulloinkin järjestettyyn kokoukseen. Tuloksena on relaatiotietokannan taulu, jossa taulun kuhunkin riviin syötetään jäsenen sekä kokouksen yksilöivä id.

Otin päätavoitteeksi luoda tietokantaan selkeän rakenteen ja normalisoisin kantaa vain tarpeelliseksi katsomani verran. Käytin apuvälineenä tietokannan suunnitteluun Fabforcen DBDesigner-ohjelman 4-versiota sekä luomiani käyttötapauskuvauksia.

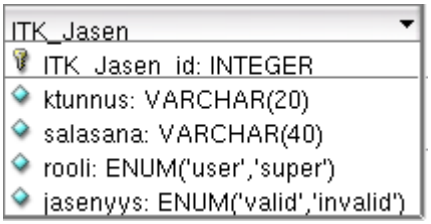
Käyttötapauskaavio (liite 1) keskittyy normaaliin käyttäjään. Normaalin käyttäjän toimenpiteisiin kuuluu sisään kirjautuminen, jolloin tarkistetaan käyttäjän rooli ja oikeudet. Pääkäyttäjän osalta siirryttäisiin pääkäyttäjän toimiin. Käyttäjältä on myös mahdollisesti lukittu pääsy tietokantaan. Normaalikäyttäjän ollessa kyseessä mahdollistetaan kirjautujalle asiaan kuuluvat toimintoja. Näitä toimintoja ovat omien tietojen muokkaaminen ja katselu, kokoukseen ilmoittautuminen ja hallitustietojen sekä vuosikokoukseen liittyvien tietojen katselu. Jäseneksi hakeminen voisi olla sivustolla mahdollista ilman kirjautumista, mutta luovuin tästä ajatuksesta. Viisaampaa lienee

on, että jäseneksi hakeutuminen tehdään esimerkiksi yhdysmiehen kautta ja hakemus tehdään hänen tunnuksillaan. Mahdollisten tuotteiden tilaamisen jätän tässä vaiheessa suunnittelun ulkopuolelle.

Käyttötapauskaaviossa 2 (liite 2) on aktorina pääkäyttäjä, jolla toimintoina ovat jäsen-, hallitus- ja kokoustietojen ylläpitäminen sekä jäsenten osallistumisaktiivisuuden seuraaminen. Pääkäyttäjinä toimivat kulloisenkin hallituksen jäsenet. Jäsentietojen ylläpitäminen pitää sisällään uuden jäsenen lisäämisen, jäsenen tietojen muokkaamisen, jäsenen oikeuksien muuttamisen sekä jäsenmaksujen kirjaamisen. Hallitustietojen ylläpitoon kuuluu hallituskokoonpanon lisääminen ja sen tietojen muokkaamisen. Kokoustietoihin liittyy kokoukseen osallistuneiden lisääminen ja pöytäkirjojen liittäminen kokoukseen. Pääkäyttäjä voi tehdä hakuja liittyen jäsenen aktiivisuuteen esimerkiksi listaamalla osallistumiset kokouksiin.

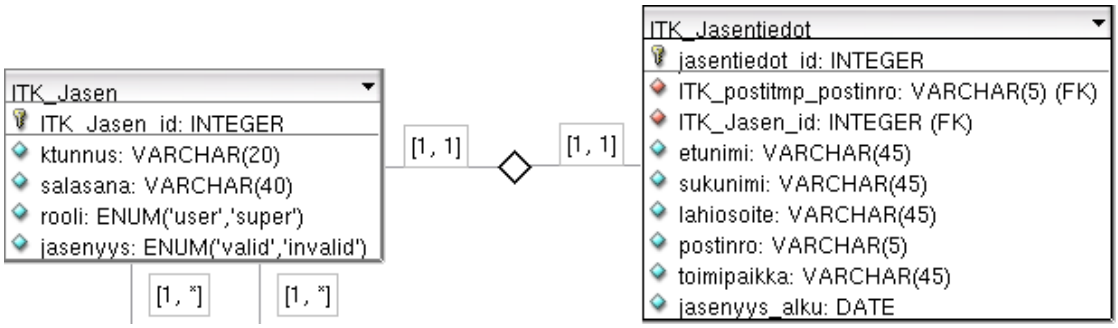
Tietokannan suunnittelu aloitetaan yleensä käsiteanalyysillä, jossa määritellään ja havainnollistetaan tietokantaan tallennettavia tietoja. Tietoja tarkastellaan aluksi loogisella ja hyvin yleisellä tasolla. Ensin määritellään käsitteet, joilla tarkoitetaan asioita tai tapahtumia, joista halutaan säilyttää tietoja tietokannassa. Näitä käsitteitä eli kohteita saadaan käyttötapausten pohjalta seuraavia: jäsen, hallitus ja kokous. Tästä saamme kohteet kolmelle taululle, joilla kullakin on oma yksilöivä pääavain. Lisäksi puhelinnumero on oma kohteensa, koska henkilöllä voi olla useita puhelinnumeroita. Puhelinnumero on heikko käsite, joka on riippuvainen toisesta käsitteestä – tässä tapauksessa käsitteestä jäsentiedot. Jäsentiedot-käsite taasen on riippuvainen jäsen-käsitteestä. Heikkoa käsitettä ei voi olla olemassa ilman ”isäkäsitettä”.

Käsitteiden määrittelyn jälkeen pohditaan niihin liittyviä ominaisuuksia eli tietoja tai attribuutteja. Esimerkiksi jäsen-käsitteen olennaisia tietoja ovat jäsenen käyttäjätunnus, käyttäjärooli ja salasana sekä lisäksi tähän yhteyteen lisäsin tiedon käyttäjän jäsenyyden voimassaolosta.



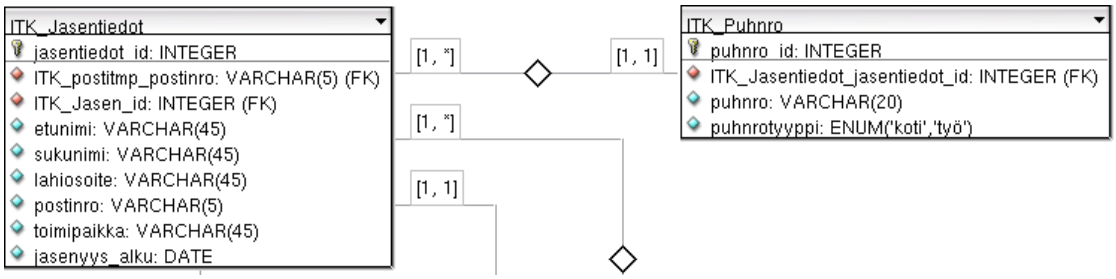
KUVIO 3. Jäsen-käsitteestä johdettu ITK_Jasen-taulu.

Loogisesti toisiinsa liittyvien käsitteiden välillä on yhteyksiä eli suhteita. Näiden käsitteiden välillä on siis liitos, joita on kolmea eri tyyppiä: yksi-yhteen, yksi-moneen ja moni-moneen. Näiden yhteyksien ja niihin liittyvien mahdollisten sääntöjen avulla pyritään säilyttämään tietokannan viite-eheys. Kuviossa 4 on esimerkki yksi-yhteen -yhteydestä. Kuviosta käy ilmi, että jäsenellä on vain yhdet jäsentiedot, jotka pitävät sisällään esimerkiksi jäsenen kotiosoitteen. Halusin eriyttää jäsentiedot omaksi käsitteekseen, koska jäsen-käsitteen koko tai lähinnä attribuuttien määrä olisi kasvanut muutoin hyvin suureksi. Yksi-yhteen -yhteydessä ensimmäisen käsitteen yksittäinen rivi liittyy vain yhteen toisen käsitteen riviin ja toisen käsitteen rivi liittyy vain yhteen ensimmäisen käsitteen riviin.



KUVIO 4. Yksi-yhteen -yhteys

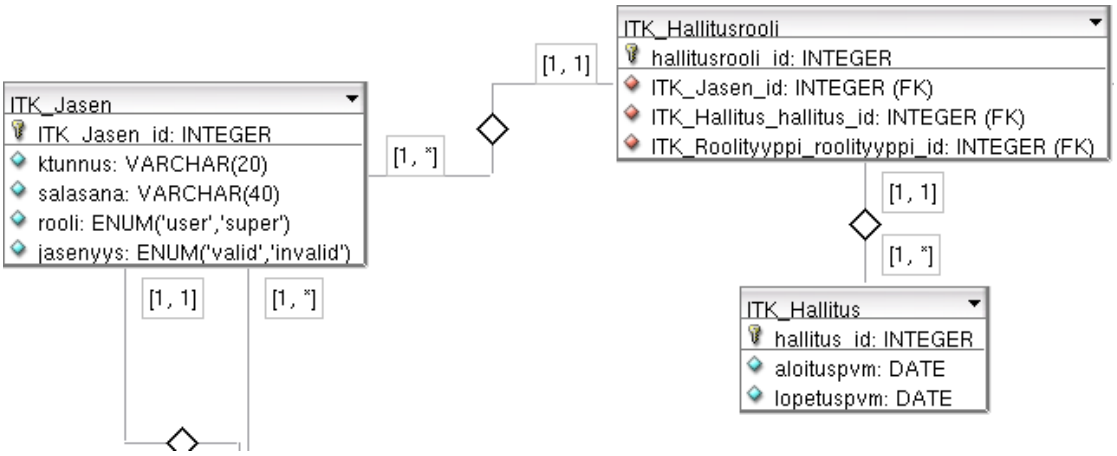
Yksi-moneen -yhteydestä on esimerkki kuviossa 5. Tällöin ensimmäisen käsitteen yksittäinen rivi voi liittyä toisen käsitteen yhteen tai useampaan riviin mutta toisen käsitteen yksittäinen rivi voi liittyä vain yhteen ensimmäisen käsitteen riviin. Kuviosta käy ilmi, että jäsenellä voi olla monta puhelinnumeroa, mutta puhelinnumero voi liittyä vain yhteen jäseneen.



KUVIO 5. Yksi-moneen -yhteys

Moni-moneen -yhteydessä ensimmäisen käsitteen yksittäinen rivi voi liittyä toisen käsitteen yhteen tai useampaan riviin ja toisen käsitteen yksittäinen rivi voi liittyä ensimmäisen käsitteen yhteen tai useampaan riviin. Moni-moneen -yhteyksiä ei sallita relaatiotietokannoissa, joten ne puretaan lisäämällä väliin assosiatiivinen käsite. Al-

kuvaiheessa tässäkin työssä oli yksi moni-moneen -yhteys, jonka lopullisessa ER-kaaviossa purin auki. Moni-moneen -yhteys esiintyi hallitus-käsitteen ja jäsen-käsitteen välillä, joiden välille jouduin lisäämään uuden käsitteen.



KUVIO 6. Jäsen- ja hallituskäsitteen yhdistäminen

Käsiteanalyysin pohjalta saamme alustavan ER-kaavion. ER-kaavion laatimisen jälkeen siirryttäisiin tarveanalyysiin, jossa käsitemallia testattaisiin ja tarkennettaisiin. Tässä työssä käsitteiden määrä on jo alkuun hyvin vähäinen, joten varsinainen tarveanalyysi jäi tekemättä. Ainoana muutoksena alustavaan ER-kaavioon oli hallitus- ja jäsen-käsitteen välisen moni-moneen -yhteyden purkaminen assosiatiivisen käsitteen avulla.

Lopullisessa ER-kaaviossa on 10 käsitettä. Esittelen ne seuraavaksi pienempiin kokonaisuuksiin jaettuna. Näitä kokonaisuuksia ovat ITK_Jasen, ITK_Hallitus, ITK_Jasentiedot ja ITK_Kokous. Jäseneen liittyvät käsitteet ovat ITK_Hallitusrooli, ITK_Jasentiedot ja ITK_Osallistujat. ITK_Hallitusrooli määrittelee jäsenen roolin tietyn kauden hallituksessa jos jäsen on kuulunut hallituskokoonpanoon. ITK_Jasentiedot sisältää jäsenen yhteystiedot ja muita tietoja kuten ammattiryhmän ja jäsenyyden alkupäivämäärän. ITK_Osallistujat määrittelee jäsenen osallistumisen tiettyyn kokoukseen.

Aikaisemmin kuviossa 6 esitin ITK_Hallituksen yhteyden ITK_Hallitusrooliin. ITK_Hallitus määrittelee kulloisellekin hallitukselle tietyn ajankohdan. Alustavassa ER-kaaviossa ITK_Jasenella oli moni-moneen -yhteys ITK_Hallituksen kanssa. Purin tämän yhteyden lopulliseen ER-kaavioon ITK_Hallitusrooli-käsitteen avulla.

ITK_Jasentiedot määrittelee jäsenen tietoja kuten osoitteen, puhelinnumeron, sähköpostiosoitteen, ammattiryhmän ja postitoimipaikan. Sillä on yhteydet käsitteisiin

ITK_Puhnro, ITK_Email, ITK_Ammattiryhma ja ITK_Postitmp. ITK_Postitmp määrittelee jäsenen postitoimipaikkatiedot. ITK_Puhnro määrittelee luonnollisesti jäsenen puhelinnumerot ja vastaavasti ITK_Email sähköpostiosoitteet, joita voi molempia olla useita. ITK_Ammattiryhma määrittelee jäsenen ammattiryhmän, joka monien muiden vastaavien attribuuttien tavoin kannattaa tarjota tietojen syöttämistä varten valmiina vaihtoehtoina. Tällöin vältytään mielivaltaiselta ja omaperäisen muotoisen tiedon syöttämiseltä ja hakujen tekeminen mahdollistuu johonkin tietyn muotoiseen tietoon mahdollistaen totuudenmukaisen vastauksen saamisen.

ITK_Kokous määrittelee tietyn kokouksen, sen päivämäärän ja tyypin. Sillä on yhteys käsitteeseen ITK_Osallistujat, joka määrittelee johonkin tiettyyn kokoukseen osallistuvat jäsenet.

5.3 Lopullinen tietokanta

Ennen tietokannan luomista se tulee vielä normalisoida. Tässä opinnäytetyössä riittävä normalisointi tapahtui jo suunnitteluvaiheessa ja ER-kaaviota luotaessa.
Lopullisen ER-kaavion (liite 3) käsitteistä muodostetaan tietokannan taulut ja käsitteiden tiedoista taulujen sarakkeet sekä käsitteiden välisistä yhteyksistä viiteavaimet. Tuloksena saadaan valmis relaatiotietokanta. Yhdistyksen web-sovelluksen tietokantaan muodostui 10 taulua:

ITK_Jasen
ITK_Jasentiedot
ITK_Puhnro
ITK_Email
ITK_Postitmp
ITK_Ammattiryhma
ITK_Hallitusrooli
ITK_Hallitus
ITK_Kokous
ITK_Osallistujat

Jokainen taulu edustaa kohdetta tai tapahtumaa. Lisäksi voi olla yhdistäviä tauluja, jotka muodostuvat assosiatiivisesta käsitteestä. Tässä tietokannassa sellainen on esimerkiksi taulu ITK_Osallistujat, joka yhdistää kohteen ITK_Jasen, toiseen kohteeseen ITK_Kokous. Kohdetaulu on esimerkiksi edellä mainittu ITK_Jasen, joka edustaa jotain konkreettista eli tässä tapauksessa yhdistyksen jäsentä. Tapahtumataulut

edustavat jonain tiettynä ajankohtana tapahtuvaa asiaa, kuten taulu ITK_Kokous, joka edustaa jonain päivämääränä pidettävää kokousta.

Ehkä yleisin relaatiotietokannan taulu on data- tai operatiivinen taulu, joka sisältää informaation antamisessa käytettävää dataa kuten jäsenen yhteystietoja (Databasejournal 2011, Types of tables in Oracle). Näiden taulujen sisältö on dynaamista, sillä näiden taulujen kanssa ollaan usein vuorovaikutuksessa. Tässä toteutuksessa kyseisiä tauluja ovat ITK_Jasentiedot, ITK_Puhnro ja ITK_Email. Operatiivisiin tauluihin läheisesti liittyy vahvistustauluja, joilla vahvistetaan epäsuorasti datatauluihin syötettäviä tietoja. Esimerkiksi taulussa ITK_Ammattiryhma tarjotaan käyttäjälle valmiita arvoja, joista hän valitsee oikean ammattiryhmän. Toinen vahvistustaulu on ITK_Postitmp.

Sarakkeet

Taulujen sarakkeiden tietotyypit valitaan tallennettavan tiedon mukaan. Tässä tietokannassa toimii tietotyyppeinä integer, char, varchar, date sekä enum. Integertyyppiä käytetään esimerkiksi ITK_Jasen-taulun ITK_jasen_id-attribuutin tietotyyppinä. ID-sarake pysyy käyttäjälle näkymättömänä ja on nopeampi käsitellä kuin vaihtuvan mittaisia tekstimuotoisia käyttäjätunnuksia (Heinisuo & Rauta 2007, 263). Kyseinen sarake on taulun perusavain ja siinä käytetään lisäksi lisämäärettä auto_increment, jolloin uusien arvojen tuottaminen ei aiheuta vaivaa. Valittu ID-sarake on siis kyseisen taulun yksilöivä, yksikäsitteinen tietue ja se on näin ollen valittu taulun perusavaimeksi. Aiemmin mainitsin, että relaatiotietokannoissa viite-eheyttä ylläpidetään taulujen välisillä liitoksilla ja niihin mahdollisesti liitetyillä johdannaissäännöillä. Liitokset, toisin sanoen viitteet toteutetaan viite-avaimilla eli taulussa on yksi tai useampia attribuutteja, jotka viittaavat toisen taulun riveihin. ITK_Jasen-taulun toinen sarake, ktunnus, on käyttäjätunnus, joka on määriteltävä lisämääreillä unique ja not null.

Määreellä unique varmistetaan käyttäjätunnuksen yksilöllisyys sekä määreellä not null estetään sarakkeen jättäminen määrittelemättä. Sarake voi kuitenkin sisältää tyhjän merkkijonon, jolloin tämä mahdollisuus on estettävä muilla keinoin, esimerkiksi PHP-koodissa. Salasana-sarakkeen määrittelin vakiomittaiseksi, sillä salasana tallennetaan tietokantaan salausfunktion käsittelemänä. Tällöin tietokannan tietoja luvattomasti lukemaan päässyt ei pääse murtautumaan sovellukseen toisen tunnuksilla – ainakaan kovinkaan helposti (Heinisuo & Rauta 2007, 264). Voisin käyttää salasanan

käsittelyyn esimerkiksi viestitiivistealgoritmia SHA-1, joka on PHP:ssä ja MySQL:ssä sisäänrakennettuna ominaisuutena.

Aluksi suunnittelin käyttäväni ITK_Jasen-taulussa attribuuttien rooli ja jasenyys kohdalla tietotyyppiä bool, jonka vaihdoin kuitenkin tietokantaa tehdessäni tietotyypiksi enum. Tähän löytyi hyvät perustelut kirjasta PHP ja MySQL (Heinisuo & Rauta 2007, 264). Enum-tyypin avulla sarakkeeseen tallennettava tieto on yksiselitteinen ja selkeä. Käytettäessä aluksi suunnittelemaani bool- tai integer-tyyppiä voitaisiin sarakkeeseen syöttää monia eri lukuarvoja ilman virheilmoitusta. Tällöin PHP-sovelluksessa joutuisi määrittelemään vakioita define-funktion avulla tai muuten suorittamaan tarkistuksia käyttäjän syötteelle. Enum-tyypin käyttö vähentää sovelluksen ylläpitoon kuluvaa vaivaa. Samaa tyyppiä voi käyttää jäsenyyden ylläpidon seuraamiseen liittyvään jasenyys-attribuuttiin.

Muiden taulujen tietotyypit valitsin samoin periaattein kuin ITK_Jasen-taulunkin. Viite-eheydestä on huolehdittava ja siinä voi käyttää apuna tietokannan viitesääntöjä. Käytin lähinnä sääntöä: ON DELETE CASCADE. Ensimmäisellä säännöllä poistetaan viittaavan taulun rivi, jos viitatun taulun viitattu rivi poistetaan. Esimerkiksi ITK_Jasentiedot-taulun luontilauseessa (liite 5) on kaksi vierasavainta; vierasavaimella ITK_Jasen_id on viitesääntö ON DELETE CASCADE. Mikäli jäsen poistetaan jäsentaulusta, poistetaan myös häneen liittyvät jäsentiedot. Vierasavaimella ITK_postitmp_postinro on viitesääntö ON UPDATE CASCADE. Jos postitoimipaikkaan liittyvässä taulussa muuttuu postinumeroniin muutos päivitetään myös jäsentiedot-tauluun. Lopussa kerrotaan taulun tyypiksi InnoDB, joka vaaditaan käytettäessä viitesääntöjä. Huomioituani viite-eheyssäännöt ja tarkistettuani tietotyyppien oikeellisuus ja tarkoituksenmukaisuus oli tietokanta valmis luotavaksi.

Tietoturvan huomioiminen tietokannassa

Edellisestä kappaleesta kävi ilmi, ettei salasanan tallentaminen selväkielisenä tietokantaan ole kovinkaan järkevä ajatus. Viestitiivistealgoritmia käytettäessä tietokantaan tallennetaan pelkkä tiiviste. Tätä tiivistettä ei ole mahdollista palauttaa salasanaksi, ja salasanaa ei siis ole tallessa missään – paitsi ehkä käyttäjän näppäimistön alla. SHA-1-tiiviste on aina 40 merkkiä pitkä, jolloin salasana-sarakkeenkin tulee olla samanmittainen.

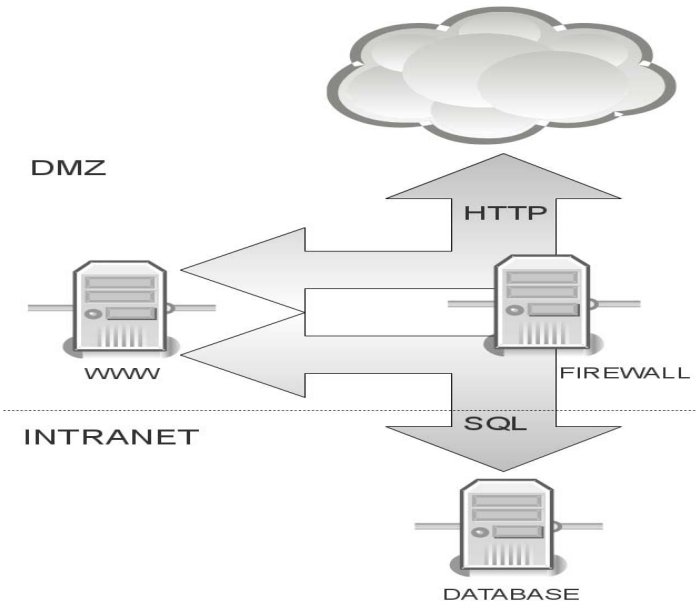
SHA1-algoritmistakin on löydetty heikkouksia, mutta se on vielä suhteellisen turvallinen. Saman viestitiivisteen voi laskea useasta ja myös erimittaisista merkkijonoista. Tällaisten päällekkäisyyksien eli törmäysten tuottaminen nopeasti vaatii kuitenkin valtavasti tehoja (Heinisuo & Rauta 2007, 266).

Merkkijono	SHA1-tiiviste
sarvikuono	1f1599d089068869b68d5e27c6fbb5c32fb31373
topparoikka	e8714bd83f751b35211f51960fbc1e3a5048ac24

KUVIO 7. Esimerkki SHA1-tiivisteestä

6 TOTEUTUS

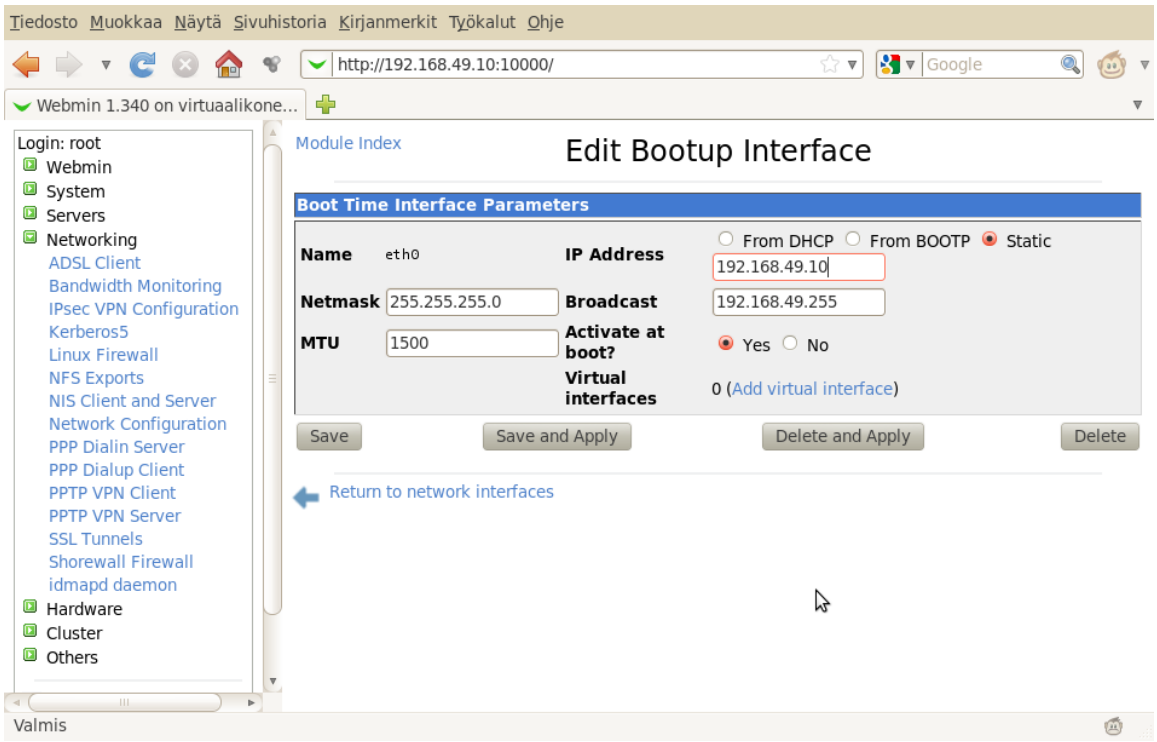
Edellisessä luvussa loin karkean suunnitelman ja pohjan sivuston toteuttamiselle. Kuten olen aikaisemmin jo maininnut valitsin perusratkaisuksi LAMP-ympäristön. Testiympäristön ja samalla mahdollisen tuotantoympäristön rakensin suhteellisen tietoturvalliseksi pidetyn ratkaisumallin mukaan eli loin kolmella virtuaalikoneella DMZ-alueen eli eteisverkon ja sisäverkon. Eteisverkon toinen virtuaalikone toimi palomuurina internetin, DMZ-alueen ja sisäverkon välillä. WWW-palvelin sijaitsi DMZ-alueella ja tietokantapalvelin sisäverkossa. Virtuaalikoneissa on käyttöjärjestelmänä CentOS-linux ja palomuuri on toteutettu linuxin omalla iptables-ratkaisulla. Virtuaalipalvelimien alustana käytin VMware Playerin versiota 3.1.3.



KUVIO 8. Testiympäristön rakenne

6.1 Palvelimien asennus

Asensin kannettavalle koneelleni PHP ja MySQL-kirjan mukana tulleen VMWare-virtuaalikoneen, jossa käyttöjärjestelmänä on Linuxin jakeluversio CentOS 5. Tietokantana virtuaalikoneessa oli valmiiina MySQL (versio 14.12) ja www-palvelimena Apache. PHP on viidettä versiota, jonka ytimenä on Zend Engine II (ver 2.1.0). Kopioin tätä virtuaalikonetta tarpeen mukaan ja muokkasin niistä tarvittavat palvelimet ja palomuurinkoneen. Ensimmäisenä vaihdoin palvelinten ip-osoitteet halutun mukaisiksi. Tämän voi tehdä jollain työkalulla kuten vaikka Webmin-hallintatyökalulla tai manuaalisesti.



KUVIO 9. IP-osoitteen muuttaminen Webmin-hallintaohjelmalla

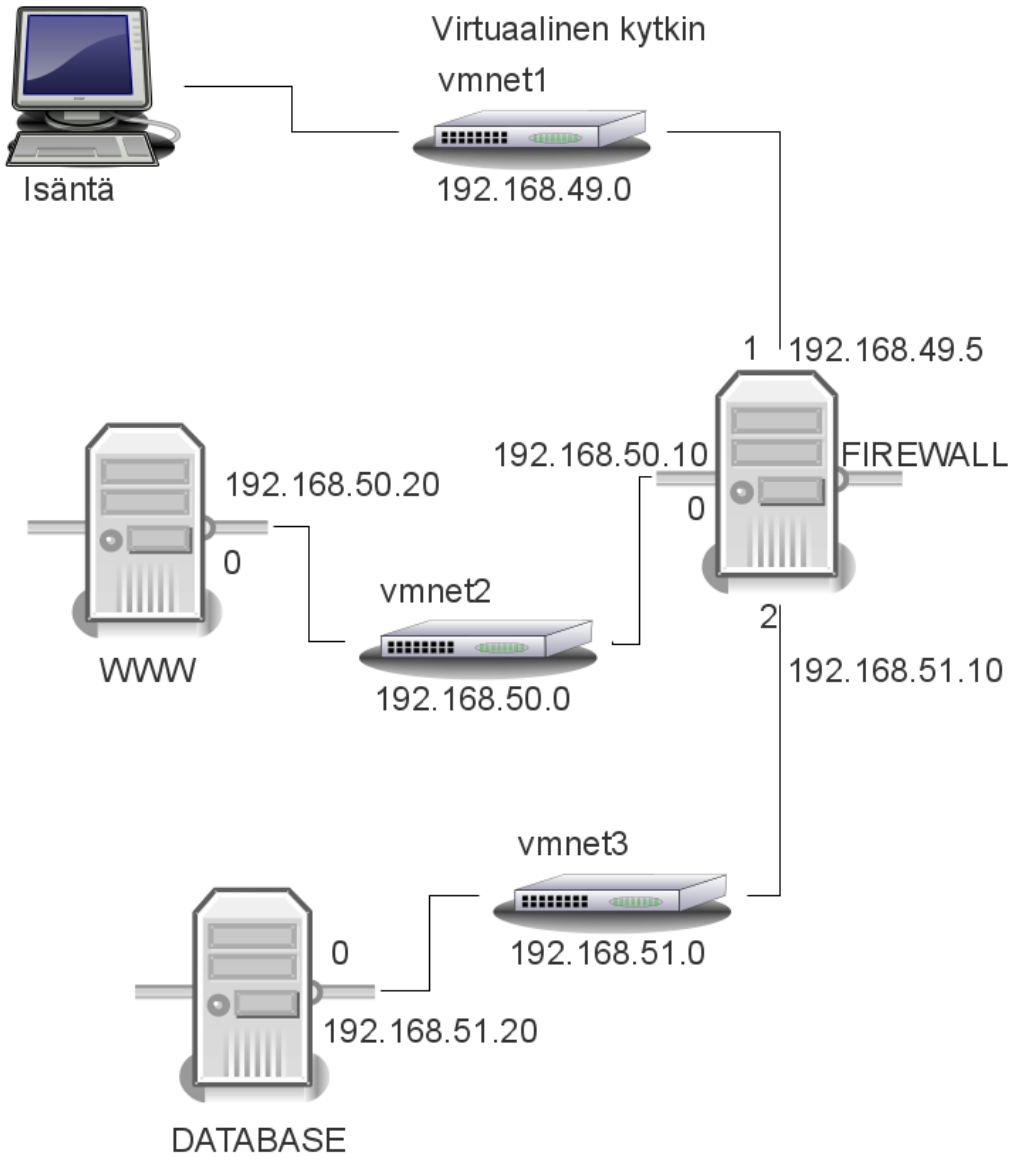
Käsin tehden ip-osoite kirjoitetaan /etc/sysconfig/network-scripts/ifcfg-eth0 -tiedostoon, kun muutos tehdään ensimmäiseen verkkokorttiin (eth0). Tämän jälkeen verkkopalvelu on käynnistettävä uudelleen komennolla:

```
/etc/init.d/network restart
```

Jos ip-osoitetta tarvitsee muuttaa vain istunnon ajaksi, se onnistuu pääkäyttäjänä yksinkertaisesti komennolla:

```
/sbin/ifconfig eth0 haluttu ip-osoite
```

Jotta voisin käyttää www-palvelinta ja tietokantapalvelinta eri aliverkoissa, jouduin lisäämään VMwaren virtuaaliverkkolaitteiden määrää oletuksesta; yksi virtuaaliverkkolaite kutakin aliverkkoa kohden (liite 4). Kuviosta 10 näkyy miten verkko on konfiguroitu; esimerkiksi www-palvelin on verkossa 192.168.50.x ja sitä varten on VMwaren virtuaaliverkkolaite (virtuaalinen kytkin) vmnet2 ip-osoitteella 192.168.50.1., joka toimii kyseisen verkon yhdyskäytävänä. Kukin VM-verkkolaite tarjoaa myös DHCP-palvelun, oletuksena osoitteesta 192.168.x.2., jos sitä haluaa käyttää.

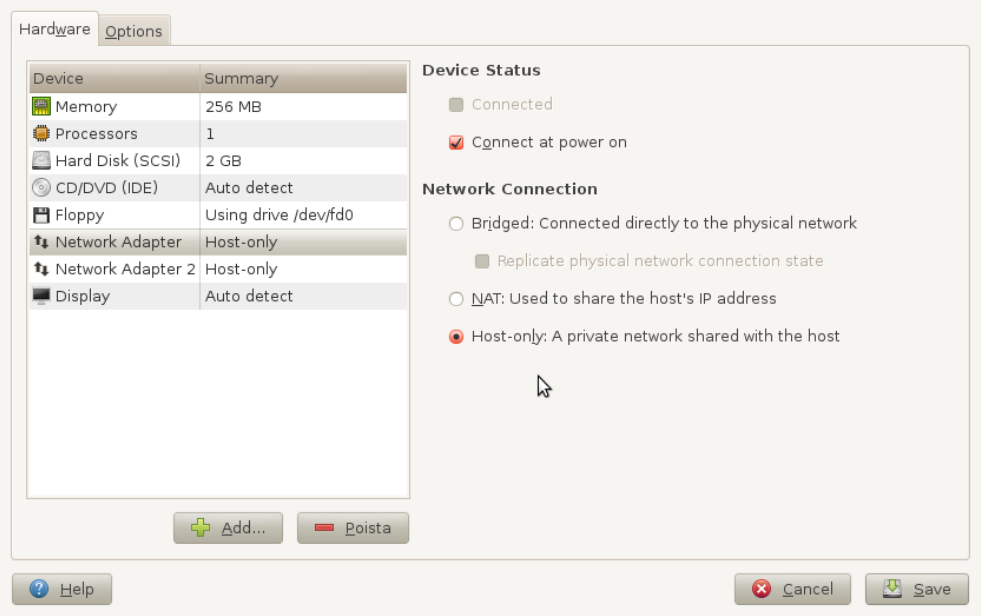


KUVIO 10. Testiympäristön IP-osoitteet

Koin ongelmia virtuaalikoneiden kytkemisessä lisättyihin VMNet-verkkolaitteisiin. Vaikka vaihdoin käsin virtuaalikoneen verkkokortin ip-osoitetta tietyn Vmnet:n osoiteavaruuteen, ei yhteyttä kuitenkaan syntynyt. VMware Playerilla en voinut valita virtuaalikoneen verkkolaitteelle muita kytkentävaihtoehtoja verkkoon kuin Bridged, NAT ja host-only.

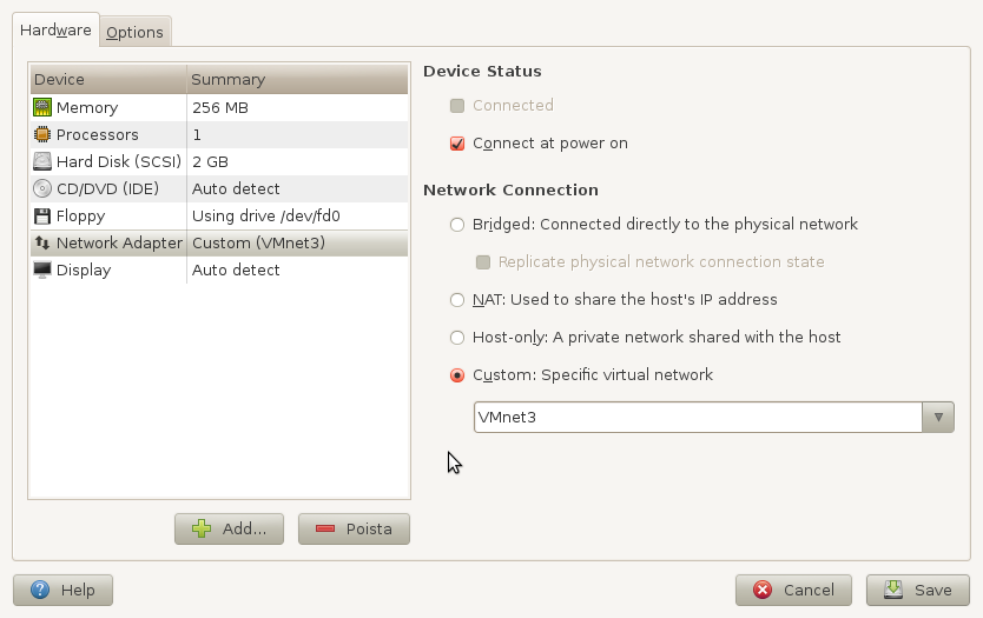
Selvisi, että host-only viittaa vain verkkolaitteeseen VMnet1, Bridged viittaa verkkolaitteeseen Vmnet0 ja NAT Vmnet8:aan. Jouduin käsin muuttamaan halutun virtuaalikoneen vmx-päätteistä tiedostoa seuraavasti:

Ethernet0.connectionType = "custom"
Ethernet0.vnet = "VMnet3"



KUVIO 9. Verkkoon kytkeytymisvaihtoehdot

Edellisen muutoksen jälkeen oli verkkoon kytkeytymisen vaihtoehtoja tullut yksi lisää: custom. Tälle vaihtoehdolle kykenin tarjoamaan haluttua VMnet-verkkolaitetta.



KUVIO 10. Uusi vaihtoehto

Muutos oli aiheuttanut virtuaalikoneen verkkokorttiin liittyviin asetustiedostoihin muutoksia ja virtuaalikone yritti kytkeytyä verkkoon DHCP:llä. Muutin jälleen käsin verkkokortin asetuksia haluamikseni /etc/sysconfig/network-scripts/ifcfg-eth0 -tiedostoon. Asetustiedoston sisältö muutosten jälkeen:

DEVICE=eth0
ONBOOT=yes
BOOTPROTO=none
HWADDR=00:0c:29:85:bb:e1
MTU=1500
NETMASK=255.255.255.0
BROADCAST=192.168.51.255
IPADDR=192.168.51.20
NETWORK=192.168.51.0
GATEWAY=192.168.51.1

Palomuurit

Seuraavaksi perehdyin palomuurien konfigurointiin; iptables-ratkaisussa iptables on käyttäjää palveleva moduuli, joka ottaa käyttäjältä tai konfigurointitiedostosta vastaan sääntöjä, joiden mukaan linuxin ytimessä oleva netfilter-moduuli käsittelee verkkolii-

kenteen ip-paketteja. Iptablesissa on kolme ennalta määriteltyä ketjua (chain), joille määritellään sääntöjä. Ketjuja ovat INPUT, jolla tarkoitetaan isäntäkoneelle päin tulevia paketteja, OUTPUT, jolla tarkoitetaan isäntäkoneelta päin lähteviä paketteja ja FORWARD niille paketeille, jotka välitetään isäntäkoneen kautta eteenpäin. Viimeisintä käytetään, jos isäntäkone toimii reitittimenä. Säännöt listataan ja verkkoliikenteen pakettia verrataan listan sääntöön alkaen listan ensimmäisestä. Mikäli paketti täsmää listassa olevaan sääntöön, kohdellaan sitä säännön mukaisesti. Tällöin pakettia ei enää verrata edempänä listassa oleviin sääntöihin. Yleensä näkökulmaksi INPUT-paketeille otetaan tapa, jossa oletuksena hylätään (DROP) kaikki liikenne ja määritellään erikseen hyväksyttävä liikenne (ACCEPT). OUTPUT-paketeille käytetään tapaa, jossa oletuksena hyväksytään kaikki lähtevät paketit ja hylätään erikseen paketit, joiden lähde on harmillinen IP-osoite tai portti, jonka takana on yksityisiä tai ei ollenkaan palveluita.

Loin tietokantapalvelimelle seuraavat palomuurisäännöt komentoriviltä käsin.
Hyväksytään kaikki sisään tuleva liikenne lo-verkkokortille (localhost):

```
/sbin/iptables -A INPUT -i lo -j ACCEPT
```

Hyväksytään kaikki aiemmin hyväksyttyyn yhteyteen liittyvä liikenne:

```
/sbin/iptables -A INPUT -m state –state ESTABLISHED,RELATED -j ACCEPT
```

Hyväksytään palomuurilta päin tuleva tcp-protokollan liikenne porttiin 3306:

```
/sbin/iptables -A INPUT -s 192.168.51.10 -i eth0 -p tcp -m tcp --dport 3306 -j ACCEPT
```

Lisätään oletussääntö (policy) tulevalle liikenteelle – hylätään kaikki:

```
/sbin/iptables -P INPUT DROP
```

Lisätään oletussääntö välitettävälle liikenteelle – hylätään kaikki:

```
/sbin/iptables -P FORWARD DROP
```

Hyväksytään oletuksena kaikki uloslähtevä liikenne:

```
/sbin/iptables -P OUTPUT ACCEPT
```

Tämän jälkeen säännöt voi tarkistaa komennolla `/sbin/iptables -L -v` ja tallentaa ne pysyvästi iptables-tiedostoon komennolla `/sbin/service iptables save` ja käynnistää palomuuripalvelun `/sbin/service iptables start`.

Toteutin vastaavat toimet www-palvelimelle huomioiden oikeat ip-osoitteet sekä portin, jota Apache kuuntelee. INPUT-sääntöihin lisäsin tietokantapalvelimen liikenteen. Ulospäin lähtevään liikenteeseen tein seuraavat muutokset:

Hyväksytään kaikki aiemmin hyväksyttyyn yhteyteen liittyvä liikenne:

```
/sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Hyväksytään ulospäin lähtevä tcp-protokollan liikenne portista 80:

```
/sbin/iptables -A OUTPUT -p tcp -m tcp --sport 80 -j ACCEPT
```

Hyväksytään ulospäin lähtevä tcp-protokollan liikenne portista 3306:

```
/sbin/iptables -A OUTPUT -p tcp -m tcp --sport 3306 -d 192.168.50.10 -j ACCEPT
```

Oletuksena hylätään kaikki ulospäin lähtevä liikenne:

```
/sbin/iptables -P OUTPUT DROP
```

Seuraavaksi siirryin määrittelemään palomuuripalvelimen sääntöjä ja ne ovatkin hieman monimutkaisemmat kuin www- ja tietokantapalvelimen säännöt. Palomuuripalvelimessa joudumme käyttämään ip-pakettien välitystä ja se mahdollistetaan muokkaamalla /etc/sysctl.conf -tiedoston erästä riviä muotoon: net.ipv4.ip_forward = 1.

Palomuuriin tein seuraavat säännöt:

Välitetään liikenne WWW-palvelimen ja tietokantapalvelimen välillä (yleisesti tarkoittaisi liikenteen välittämistä eteisverkon ja sisäverkon välillä):

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state
NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A FORWARD -i eth1 -o eth2 -m state --state ESTAB-
LISHED,RELATED -j ACCEPT
```

Välitetään liikenne www-palvelimen ja internetin välillä:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state ESTAB-
LISHED,RELATED -j ACCEPT
```

Ohjataan tuleva http-liikenne www-palvelimelle (portin välitys):

```
iptables -A PREROUTING -t nat -p tcp -i eth0 -d 192.168.49.5 --dport
80 -j DNAT --to-destination 192.168.50.20:80
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 80 -j ACCEPT
```

Ohjataan tuleva https-liikenne www-palvelimelle:

```
iptables -A PREROUTING -t nat -p tcp -i eth0 -d 192.168.49.5 --dport
443 -j DNAT --to-destination 192.168.50.20:443
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 443 -j ACCEPT
```

Ohjataan www-palvelimen liikenne tietokantapalvelimelle:

```
iptables -t nat -A PREROUTING -p tcp -i eth1 --dport 3306 -j DNAT --
to-destination 192.168.51.20:3306
iptables -A FORWARD -s 192.168.50.20 -d 192.168.51.20 -p tcp --dport
3306 -j ACCEPT
```

Mahdollistetaan www-palvelimen liikennöinti internettiin

```
iptables -A POSTROUTING -t nat -j MASQUERADE -o eth1
```

Mahdollistetaan www-palvelimen liikennöinti tietokantapalvelimelle

```
iptables -A POSTROUTING -t nat -j MASQUERADE -o eth2
```

Näillä säännöillä verkon liikennöinti toimi ajatellun mukaisesti ja pystyin siirtymään tietokannan sekä sivuston luomiseen ja testaamiseen.

6.2 Tietokannan luominen

Tietokannan loin PHPMyAdmin-ohjelmalla käyttäen DBDesigner-ohjelman valmiiksi rakentamia SQL-lauseita (liite 5). Vaihtoehtoisesti olisin voinut käyttää MySQL:n omaa komentorivipäätettä. Loin samassa yhteydessä ITK-tietokannan käyttöoikeudet käyttäjänimelle, jolla sivusto kirjautuu tietokantaan. Jätin tässä vaiheessa tarkemmat käyttöoikeuksien määrittelyt ja asettelut tekemättä. Aivan hyvin voisi jo nyt määritellä eri oikeudet normaalikäyttäjälle ja pääkäyttäjälle; esimerkiksi normaalikäyttäjä saisi päivitysoikeudet (UPDATE) vain tauluihin, joissa on häneen itseensä liittyviä yhteystietoja ja muihin tauluihin olisi vain lukuoikeudet (SELECT). Pääkäyttäjälle taasen myönnettäisiin luku-, lisäys- ja päivitysoikeudet kaikkiin tauluihin. Oikeudet lisätään MySQL:n komennolla GRANT.

CREATE USER *käyttäjätunnus@ip-osoite* IDENTIFIED BY *salasana*;
GRANT ALL PRIVILEGES ON ITK TO *käyttäjätunnus@ip-osoite*;

Edellinen SQL-lause antaa kaikki oikeudet ITK-tietokantaan tietylle käyttäjälle, joka on kirjautunut myös halutusta ip-osoitteesta. Usein ei siis ole järkevää myöntää normaalikäyttäjälle kaikkia oikeuksia, joten seuraavassa esimerkissä käyttäjälle annetaan vain SELECT-hakuoikeus tietokannan tauluihin:

GRANT SELECT ON ITK.* TO *käyttäjätunnus@ip-osoite*;

Ja toinen esimerkki, jossa annetaan lisäys- ja päivitysoikeus puhelinnumeroja sisältävään tauluun:

GRANT INSERT, UPDATE ON ITK.puhnro TO *käyttäjätunnus@ip-osoite*;

Oikeuksia poistetaan vastaavasti REVOKE-komennolla, joka muistuttaa läheisesti GRANT-komentoa. REVOKE-komennon syntaksi on seuraava:

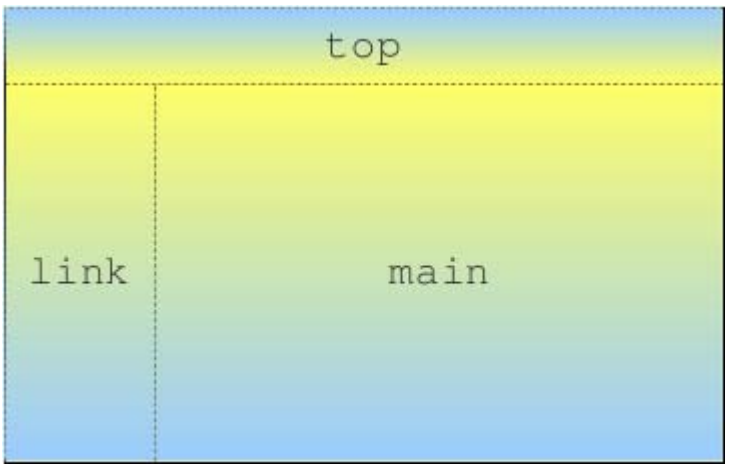
REVOKE *haluttu oikeus* ON *tietokannan nimi* FROM *käyttäjätunnus*;

6.3 Sivuston rakenne

Alkuperäisiä määrittelyitä ja tarpeita sekä niiden pohjalta luotuja käyttötapauksia tarkasteltaessa käy ilmi, ettei tuleva sivusto tulisi olemaan valtavan monimutkainen saati runsassivuinen. Sen vuoksi valitsin suhteellisen suoraviivaisen lähestymistavan sivuston rakenteen suunnitteluun ja päätin luoda käytännössä omat näkymät eli sivun

jokaiselle käyttötapaukselle. Lisäksi sivustolla tulee tehdä ero normaalin käyttäjän ja ylläpitäjän välillä. Edellisen vuoksi sivustolle tultaessa näytetään aluksi vain tervehdyssivu, jossa kerrotaan yleisesti mistä sivustolla on kyse. Alkusivulla on mahdollisuus kirjautua sisälle, ja kirjautumisen jälkeen käyttäjä ohjautuu joko normaalin käyttäjän sivuille tai ylläpitäjän sivuille. Tavallinen käyttäjä näkisi vain omat tiedot, hallituksen kokoonpanon, pöytäkirjoja ja muita asioita, joita yhdistys haluaa tuoda jäsenien tietoon. Lisäksi hän voisi täyttää ja lähettää ulkopuolisen henkilön jäsenhakemuslomakkeen. Ylläpitäjän sivusto rakentuisi mahdollisuudesta muokata jäsenien ja hallituksen tietoja sekä lisätä ja poistaa uusia käyttäjiä. Lisäksi ylläpitäjällä on mahdollisuus hallinnoida käyttäjien käyttöoikeuksia.

Sivuston tulisi ennen kaikkea olla selkeä ja helppokäyttöinen, joten päätin rakentaa sivuston kolmen elementin varaan (kuvio 11). Vakaana pysyvä otsikkoelementti ja vasemmalla reunalla sijaitseva linkkielementti, jonka linkeistä pääsee liikkumaan sisältösivuilla. Keskeisin osa näytöstä olisi sisältöelementti, jonka näkymä olisi pääasiallisesti dynaamisin ja toiminnallisin osa sivustoa. Alkuvaiheessa pääpaino olisi toiminnallisuuden ja selkeän rungon tuottamisessa sekä tietoturvan huomioimisessa. Mikäli sivusto otettaneen käyttöön, tulee jonkun perehtyä huolellisen ja luottamusta herättävän ulkoasun luomiseen.



KUVIO 11. Sivuston elementit

6.4 Sivuston toteutus

Sivustolle saapuessa on sisältöelementissä yhdistyksen lyhyt esittely ja linkkielementissä on kirjautumista varten kentät. Kirjautuessa käyttäjä ohjataan käyttäjän roolin mukaiselle sivulle. Sisältöelementtiin avautuu esimerkiksi puheenjohtajan tervehdys kuluvalle kaudelle ja linkkielementissä on tarvittavat linkit eri toimintojen mukaan.

6.4.1 Kirjautumissivu

Alkusivu kirjautumisineen on tärkeä osa sovellusta. Siinä käsitellään aluksi lomakkeella lähetetyt tiedot. PHP:ssä on käsite istunto, jonka avulla säilytetään tietoja käyttäjäkohtaisesti sivunlatauksesta toiseen ja nämä istunnot käyttävät sisäisesti evästeitä kunkin käyttäjän tunnistamiseen (Heinisuo & Rauta 2007, 273). Jos PHP-skriptissä aloitetaan istunto, lähettää palvelin käyttäjän selaimelle evästeen PHPSESSID eli ”PHP session id”. Evästeelle annetaan arvo, joka on satunnainen jono kirjaimia ja numeroita peräkkäin. Tätä merkkijonoa vastaava tiedosto sijaitsee palvelimella ja siihen tallennetaan istunnon tiedot. Tähän istuntotiedostoon tallennettuja tietoja käytetään taulukon $_SESSION kautta komennolla session_start. Istuntotaulukon arvot säilyvät korkeintaan selainistunnon ajan ja istunnon kestoon vaikuttaa evästeen elinikä sekä erikseen määritelty istuntotiedoston enimmäiselinikä. Evästeen voi tuhota esimerkiksi asettamalla sen viimeiseksi voimassaolohetkeksi menneisyydessä olevan ajankohdan ja viimeistään selaimen sulkeutuessa se tuhoutuu.

Kirjautumissivun yksinkertaistettu rakenne ja toiminnallisuus on esitetty liitteessä 7. Sivulla käsitellään käyttäjän lomakkeella lähetetyt tiedot. Jos kirjautuminen onnistuu, ohjataan käyttäjä roolin mukaiselle pääsivulle. Mikäli kirjautuminen epäonnistuu, tulostetaan sisältöelementtiin virheilmoitus. Virheiden käsittely sovelluksen toimesta tai virheilmoitusten tulostamisen estäminen on osa sivuston tietoturvaa. Oletuksena käyttäjälle näkyvät virheilmoitukset paljastavat usein turhaan tietoa sovelluksesta tai palvelimen kansiorakenteesta – tämän vuoksi asiaan kannattaa kiinnittää huomiota. Kirjautumissivu käsittelee myös käyttäjän uloskirjautumisen.

Sivun alussa sisällytetään tiedosto functions.php, jossa on sovelluksessa useasti käytettyjä funktioita:
Initialize() - ottaa istunnon käyttöön sivulla,
Check_signed_in() - selvittää, onko käyttäjä kirjautunut sisään.
Sivuston rakennetta voidaan selventää jakamalla eri tehtävät eri tiedostoihin ja sisällyttämällä näitä tiedostoja tarpeen mukaan. PHP:ssä voidaan sisällyttää tiedostoja neljällä eri komennolla:

include
require
include_once
require_once.

Jos pyydettyä tiedostoa ei löydy, lopetetaan sivun suoritus käytettäessä require-komentoa. Include-komento aiheuttaa vastaavassa tilanteessa virheen mutta sivun suoritus ei kuitenkaan keskeydy. Include_once- ja require_once-komentoja käytettäessä tiedostoa ei sisällytetä, jos se on jo aiemmin sivulle sisällytetty (Heinisuo & Rauta 2007, 288).

Sisällytetyssä tiedostossa on lisäksi funktio db_connection(), jolla luodaan yhteys tietokantaan. Tämän jälkeen siirrytään mahdollisen sisäänkirjautumisyrityksen käsittelyyn. Alla olevalla lauseella muodostetaan tietokantayhteyden sisältävä olio:

```
$ITK_connection = db_connection();
```

Funktio db_connection() palauttaa PDO-tietokantayhteysolion, jota voi kutsua useasti ilman, että yhteyttä yritetään avata montaa kertaa (Heinisuo & Rauta 2007, 375). Seuraavaksi muodostetaan staattinen muuttuja, joka ei katoa vaikka funktiosta poistutaan ja alustetaan se arvolla false. Mikäli muuttujan arvo on kuitenkin muu kuin false, lopetetaan funktion suoritus.

```
static $ITK_connection = false;
 if($ITK_connection != false)
    return $ITK_connection;
```

Jos yhteyttä ei ole vielä avattu, se avataan seuraavassa vaiheessa. Yhteyden avaamisen epäonnistuessa, välitetään virheilmoitus.

```
try
  {
    $ITK_connection = new
PDO("mysql:host=localhost;dbname=ITK",
                  "tietokannan käyttäjätunnus", "salasana");
  }
  catch(PDOException $exception)
  {
    exit("Tietokantavirhe: ".$exception->getMessage());
  }
return $ITK_connection;
```

Aikaisemmin mainitsin salasanan tallentamisesta tietokantaan viestitiivisteenä. Tämän vuoksi on laskettava käyttäjän antamasta salasanasta viestitiiviste esimerkiksi funktion sha1 avulla, jonka jälkeen tietokantahaussa verrataan näitä tiivisteitä keskenään.

Salasanaa voidaan vahventaa lisäämällä käyttäjään omaan salasanaan merkkejä (Heinisuo & Rauta 2007, 279). Täytyy muistaa, että lisättyjen merkkien tulee olla samat yhden sovelluksen sisällä. Jos mahdollinen murtautuja saa käsiinsä salasanojen tiivisteitä, hankaloituu hänen työnsä huomattavasti edellä mainitun keinon myötä. Murtautuja yrittäisi todennäköisesti laskea tiivisteitä yleisesti käytetyistä sanoista ja verrata niitä anastamiinsa salasanojen tiivisteisiin. Murtautujan yritykset kyetään käytännössä vesittämään lisäämällä salasanaan jokin hankalasti arvattava merkkijono.

```
$ITK_user_pword = sha1($_POST["password"] . "I#*T@K!TK");
```

Tiivisteen laskemisen jälkeen suoritetaan tietokantahaku valmistelluilla lausekkeilla eli käytetään hyväksi PHP:n PDO-lisäosaa. PDO-lisäosaa käsiteltiin aiemmin luvussa 4.4. Alla olevassa koodin pätkässä näkyy, miten käyttäjän syöttämä käyttäjätunnus ja salasanasta laskettu tiiviste sidotaan hakulausekkeen parametreihin.

```
$sql = "select ITK_Jasen_id, ktunnus, salasana, rooli, jasenyys
      from ITK_Jasen
      where ktunnus = :tunnus and salasana = :salasana";
$sql_clause = $ITK_connection->prepare($sql);
if($sql_clause)
{
  $sql_clause->bindParam(":tunnus", $_POST["username"]);
  $sql_clause->bindParam(":salasana", $ITK_user_pword);
}
```

Jos haku suoritettiin onnistuneesti, aloitetaan istunto session_start-funktiokutsulla. Session_start-funktiota käytetään istunnon aloittamiseen ja jatkamiseen. Kyseinen funktio lähettää käyttäjän selaimelle evästeen ja luo palvelimelle istuntotiedoston. Istuntotiedostoon tallennetaan käyttäjän perustiedot; id ja käyttäjätunnus. Tämän jälkeen käyttäjä ohjataan roolinsa mukaiselle pääsivulle.

```
  session_start();
  $_SESSION["ITK_user_id"] = $sql_result["ITK_Jasen_id"];
```

```
$_SESSION["ITK_username"] = $sql_result["ktunnus"];

reroute("user/ITK.php");
```

Kirjautumissivu käsittelee myös käyttäjän uloskirjautumisen. Käyttäjän halutessa poistua sovelluksesta, hän napauttaa linkkiä, jonka osoitteena on:

index.php?logout=true.

Tällöin sovellus tietää GET-parametrista, että käyttäjä haluaa kirjautua ulos. Ensin ladataan istunto käyttöön kutsumalla session_start-funktiota. Eväste tuhotaan asettamalla sen voimassaoloaika käsin menneisyyteen. Aluksi selvitetään evästetaulukosta $_COOKIE evästeen nimi funktiolla session_name. Tämän jälkeen voidaan funktiolla setcookie lähettää eväste HTTP-vastauksessa selaimelle ja antaa sen parametreille haluttuja arvoja – tässä tapauksessa muuttuneen voimassaoloajan. Voimassaoloajan osoittaessa menneisyyteen tuhoaa selain välittömästi evästeen.

```
if (isset($_COOKIE[session_name()]))
  setcookie(session_name(), '', time()-42000, '/');
```

Seuraavaksi tuhotaan istuntodata palvelimelta kutsumalla funktiota session_destroy. Näiden vaiheiden jälkeen ei ole tallella mitään tietoa, jonka mukaan käyttäjä olisi kirjautuneena sovellukseen.

6.4.2 Tavallisen käyttäjän pääsivu

Käyttäjän roolin mukaisen pääsivun alussa sisällytetään jälleen functions.php-tiedosto sekä tiedosto ITK.js, joka sisältää sivuston tarvitsemat javascript-funktiot. Funktiolla Initialize() otetaan istunto käyttöön ja funktiolla Check_signing() tarkistetaan onko käyttäjä kirjautunut sisään. Käyttäjä ohjataan kirjautumissivulle, jos hän ei ole kirjautunut sisään. Tällä funktiolla estetään sivuille pääseminen ilman kirjautumista. Hyödynsin kirjan PHP ja MySQL (Heinisuo & Rauta 2007) mallia AJAX-tekniikan käytöstä. Käyttäjän valitessa jonkin eri toiminnon tai tehtävän link-elementtiä klikkaamalla, kutsutaan javascript-funktiota, joka lataa valinnan mukaisen erillisen PHP-tiedoston. Tämä PHP-tiedosto ladataan pääsivun haluttuun div-elementtiin, tässä tapauksessa elementtiin main. Tiedoston lataamisen tai sisällyttämisen yhteydessä tarkastetaan haetaanko tiedostoa oikeaoppisesti eli pääsivun sisällyttämänä. Jos tiedostoa haetaan suoraan, suoritus keskeytetään. Tähän käytetään funktiota

get_included_files. Se palauttaa taulukon, johon on listattu kaikki include- ja require-funktioiden avulla sivulle lisätyt tiedostot. Taulukon koon avulla voidaan tarkastaa, onko sivua kutsuttu suoraan vai oikeaa reittiä pitkin. Tiedoston käyttäminen suoraan voitaisiin estää myös sijoittamalla se palvelimella paikkaan, josta sitä ei voi kutsua suoraan selaimella tai kansioon, joka on suojattu Apachen asetuksilla (Heinisuo & Rauta 2007, 296).

6.4.3 Tavallisen käyttäjän jäsentiedot-sivu

Esimerkkinä tavallisen käyttäjän sivuilla on linkki-elementissä toiminnallinen linkki, josta käyttäjä avaa sivun, jossa hän näkee osan omista jäsentiedoistaan ja pääsee muuttamaan niitä (liite 8).

```
echo "<div class=\"lisaa\" on-
click=\"get_member_information($_SESSION[ITK_user_id])\">Omat tiedot</div>";
```

Käyttäjä kutsuu yllä olevaa elementtiä napauttamalla funktiota get_member_information, jolla on parametrina käyttäjän id-tunniste. Funktiossa tallennetaan muuttujaan get_page osoite, jota halutaan kutsua www-palvelimelta. Tämän jälkeen kutsutaan funktiota get_XML parametreilla get_page ja replace_element.

```
function get_member_information(ITK_user_id)
{
  var get_page =
"../included/get_xml.php?ITK_user_id="+ITK_user_id+"&get_member_information=tr
ue";
get_XML(get_page, replace_element);
}
```

Funktion get_XML saama parametri get_page kertoo, mihin osoitteeseen pyyntö tehdään ja toisena parametrina on funktion nimi, jota kutsutaan sen jälkeen, kun palvelin on palauttanut vastauksensa selaimelle (Heinisuo-Rauta s.334).

```
var request;
function get_XML(get_page, handler) {
request = new XMLHttpRequest();
   request.onreadystatechange = handler;
   request.open("GET", get_page, true);
```

```
request.send(null); }
```

Muuttujaan request tallennetaan XMLHttpRequest-olio. Se on toteutettu Microsoftin ActiveX-tekniikan avulla. Olion onreadystatechange-jäsenmuuttujan arvoksi asetetaan parametrina saatu funktio. Kyseistä funktiota kutsutaan sen jälkeen, kun selain on saanut dokumentin vastaukseksi palvelimelta. Seuraavaksi kutsutaan olion jäsenfunktiota open. Sen parametreja ovat ”GET”-pyyntö ja www-sivun osoite, jonka get_XML-funktio sai parametrinaan. Kolmas parametri true kertoo, että pyyntö on asynkroninen. Asynkronisessa pyynnössä funktiosta palataan välittömästi pyynnön jälkeen ja sitä käsitellään automaattisesti taustalla. Kaiken kaikkiaan funktio get_XML luo vain uuden olion, jonka avulla tehdään pyyntö määriteltyyn osoitteeseen.

Funktio replace_element, jota get_XML-funktio kutsuu, on hieman monimutkaisempi. Se käsittelee request-muuttujaa. Request-muuttujassa olevalla oliolla on readyState-jäsenmuuttuja, jonka avulla voidaan seurata missä vaiheessa pyyntö on menossa. Kun jäsenmuuttuja saa arvon 4, on pyyntö suoritettu loppuun. Jäsenmuuttuja status sisältää palvelimen palauttaman tilakoodin. Kun tilakoodi saa arvon 200, on pyynnön kohteena ollut osoite löytynyt. Oliolla on jäsenmuuttuja responseXML, joka sisältää pyynnöllä haetun dokumentin XML-rakenteena. Toinen vastaavanlainen jäsenmuuttuja on responseText, joka sisältää palautetun dokumentin tavallisena tekstinä. Tässä tapauksessa käytetään ensin mainittua jäsenmuuttujaa. Tämän muuttujan jäsenfunktiolla getElementsByTagName haetaan noudetun XML-dokumentin halutut elementit, jotka sijoitetaan taulukon tavoin toimivassa kokoelmassa muuttujaan replaceables.

```
var new_contents =
  request.responseXML.getElementsByTagName("container");
```

Muuttujaan new_contents sijoitetut container-elementit käydään läpi for-silmukalla poimien noudetun dokumentin määrittelemän elementin target_element-attribuutin arvo.

```
for(var i = 0; i < new_contents.length; ++i) {
                var new_content = new_contents[i];
                var new_content_id =        new_content.getAttribute("target_element");
```

Pääsääntöisesti tällä sivustolla kohde-elementtinä on main-elementti kuten alla olevassa dokumentin osassa.

```
elseif(isset($_GET["get_member_information"]))
{
  echo "<container target_element=\"main\">\n";
  require("../user/member_information.php");
  echo "</container>\n";
```

Kun korvattavan elementin tunniste löytyy, haetaan muuttujaan replaceable elementti, jonka sisältö korvataan. Jäsenfunktio getElementById palauttaa elementin, jonka id-attribuutin arvo annetaan parametrissa – tässä tapauksessa main-elementin. On huomioitava, ettei sivulla saa olla samalla tunnisteella varustettuja elementtejä. Jäsenfunktio getElementById palauttaa vain yhden elementin.

```
var replaceable = document.getElementById(new_content_id);
```

Tämän elementin kaikki lapsielementit poistetaan ja lisätään tilalle korvattavat elementit. Javascriptissä elementtejä käsitellään aina viittauksina eli esimerkissä muuttujassa replaceable ei ole kopiota elementistä vaan viittaus document-olion sisällä olevaan elementtiin (Heinisuo & Rauta 2007, 337). Jos muuttujaan sijoitetaan toinen elementti, vaihdetaan siinä ollut viittaus viittaukseksi uuteen elementtiin.

Muuttujan replaceable jäsenfunktio hasChildNodes kertoo, onko elementillä lapsielementtejä. Lapsielementtejä poistetaan, kunnes niitä ei enää ole:

```
 while(replaceable.hasChildNodes())
        replaceable.removeChild(replaceable.firstChild);
```

Lapsielementtien poiston jälkeen lisätään tilalle uusi korvaava elementti. Kaikilla elementeillä on childNodes jäsenfunktio, joka sisältää kaikki elementin lapset. Tällä jäsenmuuttujalla on length-jäsenmuuttuja, joka kertoo lapsielementtien määrän. Kukin lapsielementti kopioidaan sivulle replaceable-muuttujan viittaman elementin sisälle:

```
  for(j = 0; j < new_content.childNodes.length; ++j)      {
    var element = new_content.childNodes[j];
    if(element.xml)
      replaceable.innerHTML += element.xml;
```

```
    else
      replaceable.appendChild(element.cloneNode(true));
```

Kun yhden container-elementin lapsielementit on kopioitu sivulle, aloitetaan ulommalla silmukalla uusi kierros, kunnes kaikki container-elementit on käyty läpi. Ulomman silmukan käsittelyn päätyttyä, on noudetusta XML-dokumentista kopioitu container-elementtien sisältö vastaaville paikoille sivulla. Esitelty menetelmä päivittää div-elementtien sisältöä viehättää itseäni. Olen aiemmin käyttänyt paljon HTML:n kehyksiä (frames), mutta niiden käyttö on vähentynyt ja ne ovatkin jäämässä pois tulevaisuuden HTML:n versioista (W3schools, HTML Frames).

Noudetun XML-dokumentin sisältö sisällyttää main-elementtiin member_information.php -tiedoston, joka näyttää kyseiselle käyttäjälle osan hänen tiedoistaan kuten nimen ja yhteystiedot (liite 8). Tietojen hakeminen halutuista tietokannan tauluista tapahtuu kirjautumissivun tapaan. Ensin tarkistetaan tietokantayhteyden olemassaolo ja sen jälkeen luodaan haluttu sql-lauseke. Käytin tietojen haussa SQL:n LEFT JOIN-ominaisuutta, jolloin voidaan hakea samalla kyselyllä tietoja useasta taulusta. LEFT JOIN antaa ensimmäisestä; vasemmanpuolimmaisesta taulusta kaikki rivit, vaikka niihin liittyviä tietoja ei olisi muissa liitetyissä tauluissa (MySQL Reference Manual, LEFT JOIN).

```
$ITK_connection = db_connection();
  $sql = "SELECT etunimi, sukunimi, lahiosoite, postinro, toimipaikka, jasenyys_alku,
ammattiryhman_selite, email_tyyppi, email, puhnro, puhnrotyyppi FROM
ITK_Jasentiedot
LEFT JOIN ITK_email ON ITK_Jasentiedot.email_id = ITK_email.email_id
LEFT JOIN ITK_puhnro ON ITK_Jasentiedot.puhelinnro_id =
ITK_puhnro.puhelinnro_id
LEFT JOIN ITK_ammattiryhma ON ITK_Jasentiedot.ammattiryhma_id =
ITK_ammattiryhma.ammattiryhma_id
WHERE jasen_id = :ITK_user_id;";

  $sql_clause = $ITK_connection->prepare($sql);
  if($sql_clause) {
    $sql_clause->bindParam(":ITK_user_id", $_REQUEST["ITK_user_id"]);

    if($sql_clause->execute())
      $info = $sql_clause->fetch();
```

Tietokantahaun onnistuessa voidaan noudetut arvot sijoittaa sivulle $info-muuttujan avulla. Mikäli käyttäjä haluaa päivittää tietojaan, hän voi muuttaa niitä tekstikentässä. Päivitettyjen tietojen lähettäminen tapahtuu napauttamalla lomakkeen Tallenna-painiketta. Lomakkeen alussa määritellään onsubmit-metodi, jonka kutsuma funktio suoritetaan lomakkeen tietoja lähetettäessä. Onsubmit kutsuu save_info-funktiota, jolle annetaan parametrina käyttäjän id-tunnus. Valitsemallani toteutustavalla en voinut käyttää normaalia lomakkeen tietojen lähetystapaa POST- tai GET-metodeilla yhdessä lomakkeen action-toiminnon kanssa. Jouduin hakemaan save_info-funktiossa jokaisen tekstikentän arvot muuttujiin, joiden avulla välitin arvot uudelle main-elementin sisällölle URL-osoitteessa. Esimerkiksi tekstikentän member_givenname arvon hakeminen muuttujaan etunimi:

```
var etunimi = document.getElementById("member_givenname").value;
```

Edellisen arvon välittäminen get_page-muuttujan sisältämässä URL-osoitteessa:

```
var get_page = "../included/get_xml.php?ITK_user_id="+ ITK_user_id
+"&update_member_information=true&etunimi="+ etunimi;
```

Tämän jälkeen haetaan main-elementtiin uusi sisältö (update_member_information.php) aiemmin tutuksi get_xml-funktiolla. Arvot, jotka välitettiin URL-osoitteessa, saadaan käyttöön esimerkiksi REQUEST-taulukon avulla. Tässä vaiheessa on myös tarkistettava käyttäjän antamat syötteet. Jo PDO-lisäosan käyttäminen lisää turvallisuutta kuten luvussa 4.4 mainitsin. Aluksi sijoitin välitetyt arvot alustettuihin muuttujiin selkeyden vuoksi, jonka jälkeen tein edellisen sivun tapaan sql-lauseen, johon sidoin parametreina edellä mainitut muuttujat. Esimerkiksi REQUEST-metodilla haetun arvon sijoittaminen muuttujaan:

```
$etunimi = $_REQUEST["etunimi"];
```

Saman muuttujan sitominen sql-lauseeseen:

```
$sql_clause->bindParam(":etunimi", $etunimi);
```

Itse sql-lauseen toteutin myös hyödyntämällä LEFT JOIN -ominaisuutta, jolloin tietojen päivittämisessä selviää yhdellä lauseella:

```
UPDATE ITK_Jasentiedot
LEFT JOIN ITK_email ON ITK_Jasentiedot.email_id = ITK_email.email_id
LEFT JOIN ITK_puhnro ON ITK_Jasentiedot.puhelinnro_id =
ITK_puhnro.puhelinnro_id
SET ITK_Jasentiedot.etunimi = :etunimi, ITK_Jasentiedot.sukunimi = :sukunimi,
---poistettu loput---
WHERE jasen_id = :ITK_user_id;
```

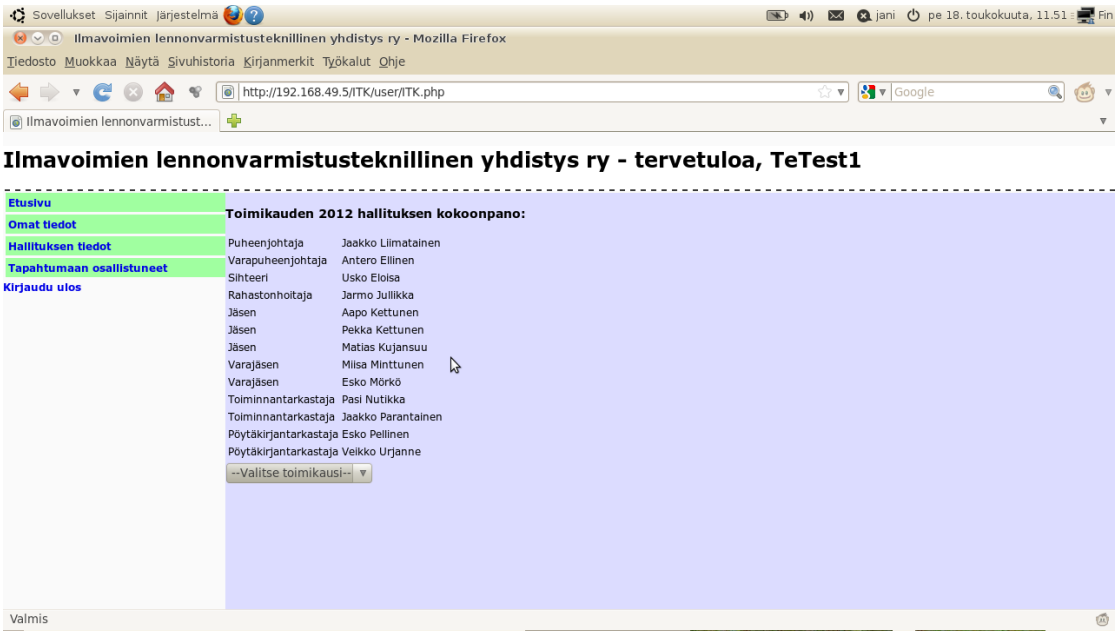
Vastaavaa tekniikkaa käyttäen jäsen voi valita sivun, johon luetellaan hallituksen kokoonpano.

6.4.4 Tavallisen käyttäjän hallituksen kokoonpano-sivu

Hallituksen kokoonpanon esittävälle sivulle tulostetaan oletuksena viimeisimmän hallituksen jäsenien nimet ja hallitusrooli. Käytin select-lauseen muodostamisessa hyväksi hallituksen kokoonpanon vakiomuotoisuutta. Select-lauseessa järjestetään haetut nimet haluttuun järjestykseen case-rakenteen avulla:

```
SELECT etunimi, sukunimi, roolityyppi FROM ITK_Jasentiedot
LEFT JOIN ITK_Hallitusrooli ON ITK_Hallitusrooli.ITK_Jasen_id =
ITK_Jasentiedot.ITK_Jasen_id
LEFT JOIN ITK_Hallitus ON
ITK_Hallitus.hallitus_id = ITK_Hallitusrooli.ITK_Hallitus_hallitus_id
WHERE ITK_Hallitus.hallitus_id = (SELECT hallitus_id FROM ITK_Hallitus WHERE
YEAR(aloituspvm) = :ITK_requested_board)
ORDER BY (CASE ITK_Hallitusrooli.roolityyppi
    WHEN 'puheenjohtaja' THEN 1
    WHEN 'varapuheenjohtaja' THEN 2
    WHEN 'sihteeri' THEN 3
    WHEN 'rahastonhoitaja' THEN 4
    WHEN 'jasen' THEN 5
    WHEN 'varajasen' THEN 6
    WHEN 'toiminnantarkastaja' THEN 7
    WHEN 'poytakirjantarkastaja' THEN 8
    END)
```

Hallituksen koostumuksen ollessa vakiomuotoinen, voidaan tietyn roolin mukaiset tiedot löytää suoraan tulosjoukkoon numeroarvoilla osoittaen.
Sivulla on alasvetovalikko, josta käyttäjä voi valita halutun tarkasteluvuoden, jolloin painikkeella Hae hallituksen tiedot saa haettua kyseisen toimikauden hallituksen kokoonpanon sivulle.



KUVIO 12. Hallituksen kokoonpano

6.4.5 Pääkäyttäjän sivut

Pääkäyttäjän toimiin kuuluu uusien jäsenien lisääminen ja tietojen muuttaminen, jäsenien oikeuksien muuttaminen ja tilien lukitseminen sekä erilaisten tapahtumien ja niihin osallistuneiden jäsenien kirjaaminen. Lisäksi pääkäyttäjä pitää yllä hallitustietoja ja -rooleja. Sivujen toteuttamiseen käytin samoja tekniikoita kuin tavallisen käyttäjän sivuilla, joten en käsittele sivujen toteuttamista enää tässä luvussa yhtä tarkasti. Omia kokonaisuuksiaan pääkäyttäjän puolella on jäsenen tietoihin liittyvät sivut, hallitukseen liittyvät sivut ja tapahtumiin liittyvät sivut. Pääkäyttäjälle on oma sivu uuden jäsenen lisäämiseen (liite 9) ja tietojen muokkaaminen tapahtuu tavallisen käyttäjän sivusta hieman muokatulla versiolla. Lisäksi jäsenen tilin lukitsemiselle sekä käyttäjätason muuttamiselle on oma sivunsa (liite 10).

Hallitustietojen hallinnoimiseen liittyviä sivuja on kaksi, joista toisella lisätään uusia hallituksia ja toisella muokataan tiettyyn hallitukseen kuuluvia jäseniä (Liite 11). Li-

säksi tapahtumiin liittyvät kaksi sivua, joista toisella lisätään tapahtuma ja toisella siihen osallistuneet jäsenet (liite 12).

6.5 Syötteiden tarkistaminen

Vaikka PDO-lisäosaa käyttämällä voidaan käyttäjän syöttämiä arvoja siistiä automaattisesti ja estää näin vaarallisten syötteiden pääsyä tietokantaan on kuitenkin viisasta suorittaa syötteiden tarkistamista myös sivuston tasalla. Käyttäjähän voi huomaamattaan tarjota kelpaamatonta syötettä. Myös tietokannassa annetut rajoitteet estävät virheellisten arvojen tallentamista tietokantaan. Syötteiden tarkistamista varten on PHP:ssä tarjolla useita funktioita, joilla käyttäjän antamaa syötettä voidaan esimerkiksi arvioida säännöllisiä lausekkeita hyväksi käyttämällä. Säännöllisellä lausekkeella määritellään merkkijonon eli käyttäjän antaman syötteen muotoa rajaavat säännöt (s. 297). Esimerkiksi seuraavalla tarkistuksella

```
if(!mb_ereg('^\d\d\d\d\d$', $postinro))
```

tutkitaan, onko muuttujan $postinro arvo kelvollinen postinumero eli viisi numeroa sisältävä numerosarja, joista jokainen numero voi olla väliltä 0-9.
Vastaavalla tavalla voidaan seuraavalla tarkistuksella

```
if(!eregi("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,3})$", $email))
```

tarkistaa, onko käyttäjän tarjoama sähköpostiosoite kelvollinen muodoltaan.
Yksinkertaisimmillaan voidaan tarkistaa käyttäjän syötteitä mittaamalla merkkijonojen pituutta mb_strlen-funktiolla. Saatu arvo tarkistettaisiin esimerkiksi if-lausekkeilla tutkien sattuuko merkkijonon pituus halutulle välille.

Syötteiden tarkistaminen on oleellisen tärkeää yhden tietoturvallisuuden osatekijän, eheyden, vuoksi. Kirjoitusvirheiden tai jopa tahallisen väärän tiedon syöttäminen tietokantaan on syytä estää ja huolellinen tarkastelu mahdollisimman alkuvaiheessa on suositeltavaa. Mikäli syötteiden tarkastaminen jätetään tietokannan huolehdittavaksi, voi olla, ettei tietokannan tauluun taltioidu haluttuja tietoja. Virheiden käsittelyn unohtuessa voi käyttäjältä jäädä väärän syötteen antaminen huomaamatta ja taulun sarakkeeseen ei taltioidu mitään, jos sarakkeelle annetut rajoitteet estävät syötteen taltioinnin.

7 JOHTOPÄÄTÖKSET

Opinnäytetyön tarkoituksena oli luoda Ilmavoimien lennonvarmennusteknilliselle yhdistykselle (ILVTY ry) internet-selaimella käytettävä jäsensivusto työpaikan sisäiseen intranet-verkkoon tai vaihtoehtoisesti julkiseen internettiin. Sivuston ja sen ohjelmisto- ja teknisten ratkaisujen, kuten tuotantoympäristön, tietoturvallisuuteen tuli työssä kiinnittää huomiota. Tavoitteena oli toteuttaa helppokäyttöinen tietokantapohjainen sovellus tietojen päivittämiseen ja arkistointiin. Lisäksi ratkaisun tuli perustua huokeisiin tai ilmaisiin ohjelmistoihin.

Työn tekniseksi alustaksi valitsin LAMP-ympäristön ilmaisuuden ja omien mieltymysteni vuoksi. Perehdyttyäni ensin tietoturvallisuuteen yleensä ja sitten tietoturvallisuuden huomioimiseen LAMP-ratkaisuissa tulin yksinkertaiseen lopputulokseen. Käsiteltävien tietojen ollessa arkaluonteisia tai erityisen salassa pidettäviä tulee sivuston ja siihen liittyvien palvelinympäristöjen luominen olla todellisten ammattilaisten tai asiantuntijoiden vastuulla. Niin laaja asia ja monelle alueelle ulottuva käsite on tietoturvallisuus. Se ulottuu käyttäjän asenteista tilaratkaisujen ja kovan raudan kautta yksittäisiin tavuihin. Tämän vuoksi ehdotan yhdistykselle, että he hankkivat jäsensivuston tarvitsemat palvelut luotettavaksi havaitulta operaattorilta. Kilpailu on alalla kiristynyt ja sen tähden veloitukset esimerkiksi LAMP-ratkaisuista ovat edullisia. Sivuston luomisessa he voivat myös käyttää kaupallisia tarjoajia tai jatkaa aloittamaani työtä.

Sivusto toteutettiin toiminnallisuuksien osalta lähes täysin. Ulkoasun ja pienien yksityiskohtien toteuttaminen jää jatkotyöskentelyn varaan. Tähän mennessä tehty työ antanee yhdistykselle hyvän kuvan siitä, mitä mahdollisuuksia jäsensivusto voisi tuoda yhdistyksen toimintaan. Perehdyttyäni kattavasti tietoturvallisuuteen uskon voivani neuvoa yhdistystä heidän päätöksenteossaan. Yhdistyksen hallituksen tehtäväksi jää päättää, miten he asiassa etenevät.

LÄHTEET

Databasejournal 2011. *Oracle: Types of Tables in Oracle* [viitattu 10.10.2011] Saatavissa: www.databasejournal.com/features/oracle/article.php/3616476/Types-of-Tables-in-Oracle.htm

F-secure 2012. *Linux server security* [viitattu 23.1.2012] Saatavissa: http://www.f-secure.com/fi/web/business_fi/products/servers/solution

Hakala, M., Vainio, M. & Vuorinen, O. 2006. *Tietoturvallisuuden käsikirja.* Porvoo: WS Bookwell.

Heinisuo, R. & Rauta, I. 2007. *PHP ja MySQL Tietokantapohjaiset verkkopalvelut.* 4. uudistettu painos. Helsinki: Talentum.

Henkilötietolaki 22.4.1999/523. Finlex. Lainsäädäntö [viitattu 13.3.2011]. Saatavissa: http://www.finlex.fi/fi/laki/ajantasa/1999/19990523

ISO/IEC 17799:2005. 2005. International Organization for Standardization. [viitattu 13.11.2011]. Saatavissa: http://www.iso.org/iso/catalogue_detail?csnumber=39612

Kaario, K. 2002. *TCP/IP-verkot*. Porvoo: WS Bookwell.

MySQL Reference Manual 2012. *LEFT JOIN* [viitattu 15.1.2012]. Saatavissa: http://dev.mysql.com/doc/refman/5.0/en/left-join-optimization.html

Secmeter 2008. *Palvelunestohyökkäys* [verkkojulkaisu]. Viitattu 5.1.2012. Saatavissa: http://www.secmeter.com/palvelunestohyokkays.html

Tenable 2012. *Tenable Nessus* [viitattu 28.1.2012]. Saatavissa: http://www.nessus.org/products/nessus

VAHTI 1/2002. 2002. Tietoteknisten laitetilojen turvallisuussuositus [verkkojulkaisu]. Valtiovarainministeriö [viitattu 14.7.2011]. Saatavissa: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20020101Tietot/name.jsp

VAHTI 2/2008. 2008. *Tärkein tekijä on ihminen – henkilöstöturvallisuus osana tieto-turvallisuutta* [verkkojulkaisu]. Valtiovarainministeriö [viitattu 15.08.2011]. Saatavissa: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20080218Taereki/name.jsp

VAHTI 3/2010. 2010. Sisäverkko-ohje [verkkojulkaisu]. Valtiovarainministeriö [viitattu 15.11.2011]. Saatavissa: https://www.vahtiohje.fi/web/guest/verkon-rakenne

VAHTI 5/2003. 2003. Käyttäjän tietoturvaohje [verkkojulkaisu]. Valtiovarainministeriö [viitattu 25.4.2011]. Saatavissa: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/51027/name.jsp

W3schools 2012. *HTML Frames* [viitattu 18.1.2012]. Saatavissa: http://www.w3schools.com/html/html_frames.asp

Wikipedia 2011. *Apache HTTP server* [viitattu 30.11.2011]. Saatavissa: http://fi.wikipedia.org/wiki/Apache_%28palvelinohjelma%29

Wikipedia 2012. *Bastille Unix* [viitattu 29.1.2012]. Saatavissa: http://en.wikipedia.org/wiki/Bastille_Unix

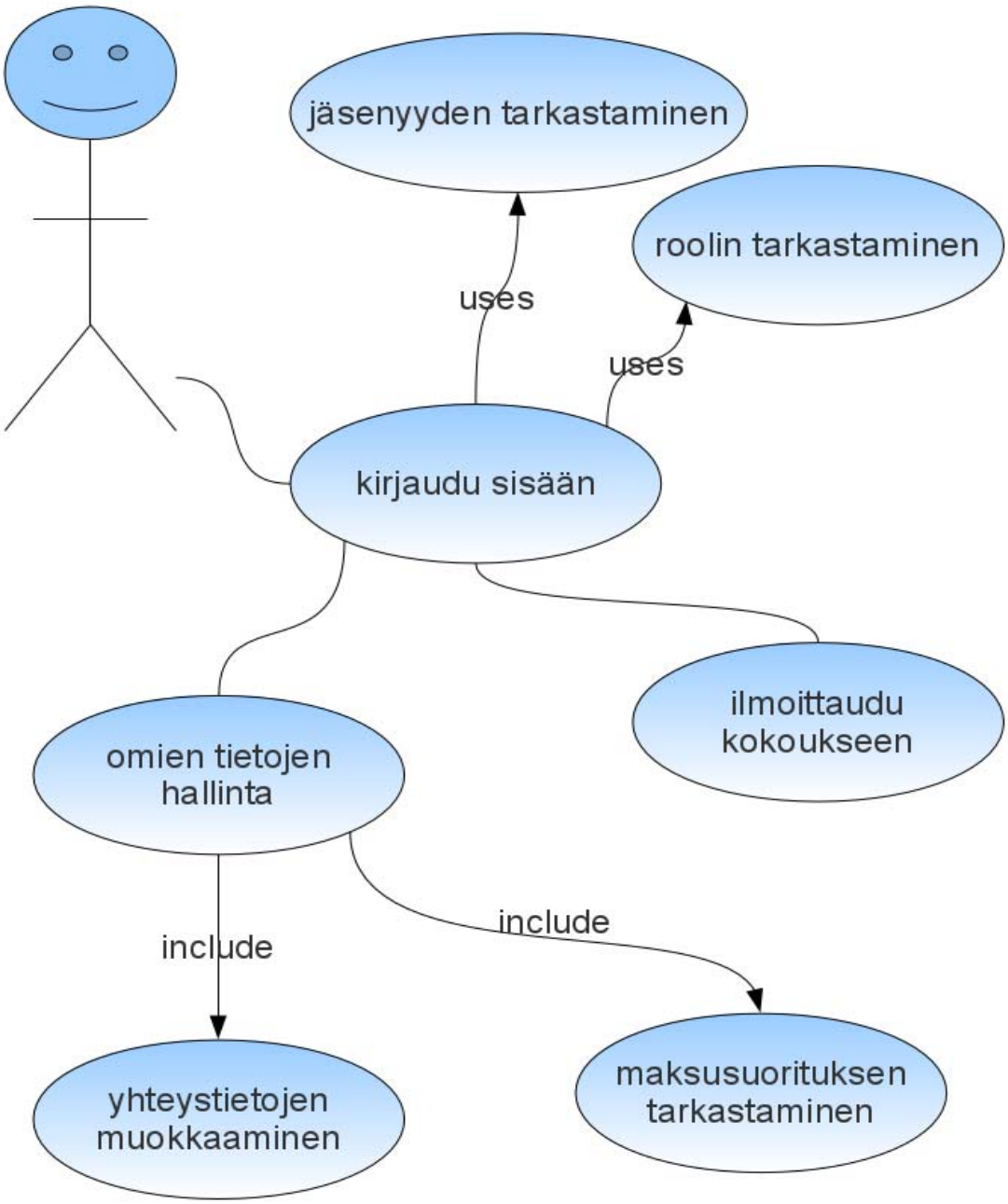
Wikipedia 2011. *Open source Tripwire* [viitattu 5.8.2011]. Saatavissa: http://en.wikipedia.org/wiki/Open_Source_Tripwire

Wikipedia 2012. *Security-Enhanced Linux* [viitattu 25.1.2012]. Saatavissa: http://en.wikipedia.org/wiki/Security-Enhanced_Linux

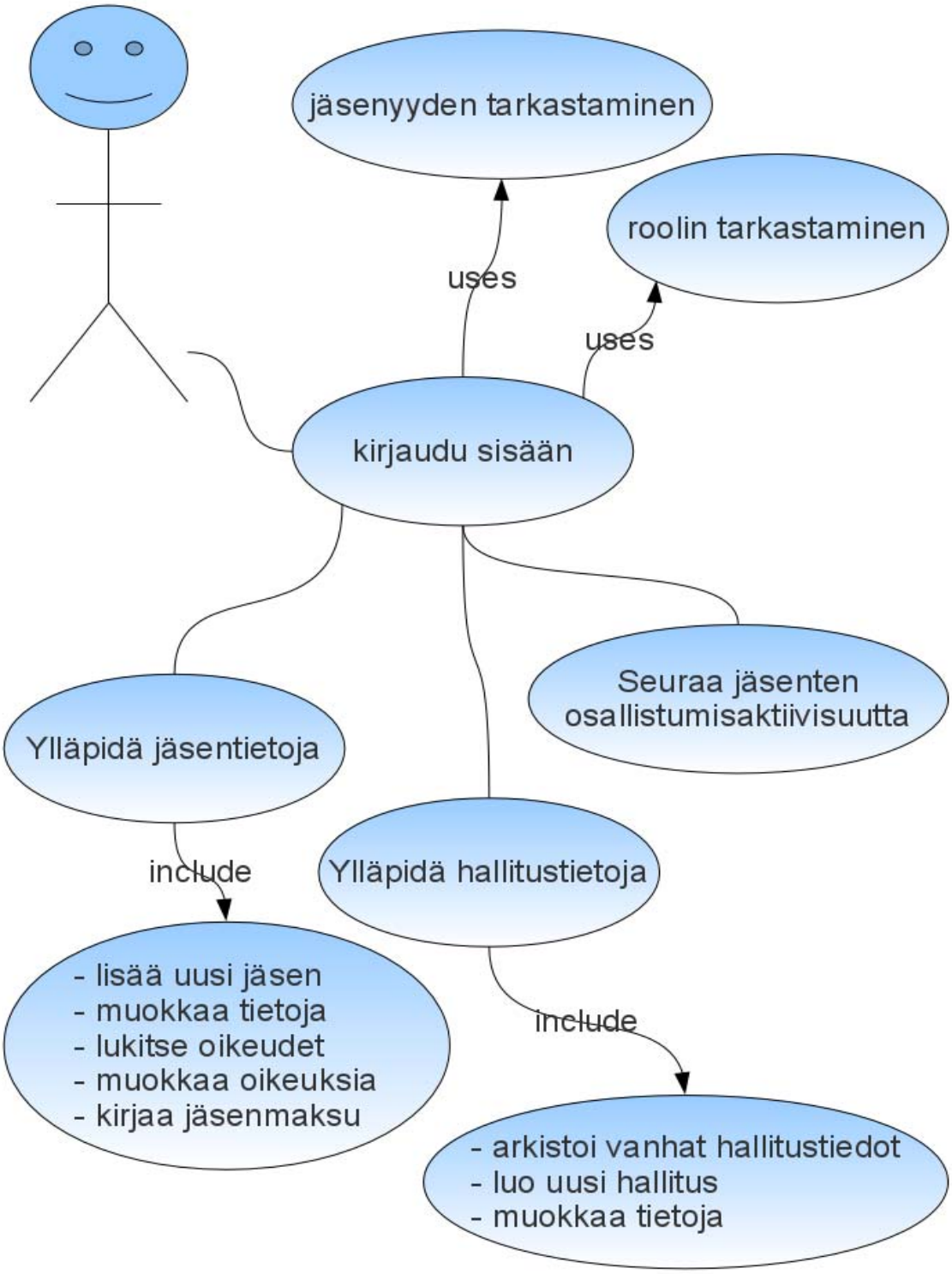
Wikipedia 2011. *SNORT* [viitattu 29.1.2012]. Saatavissa: http://fi.wikipedia.org/wiki/SNORT

Liite 1

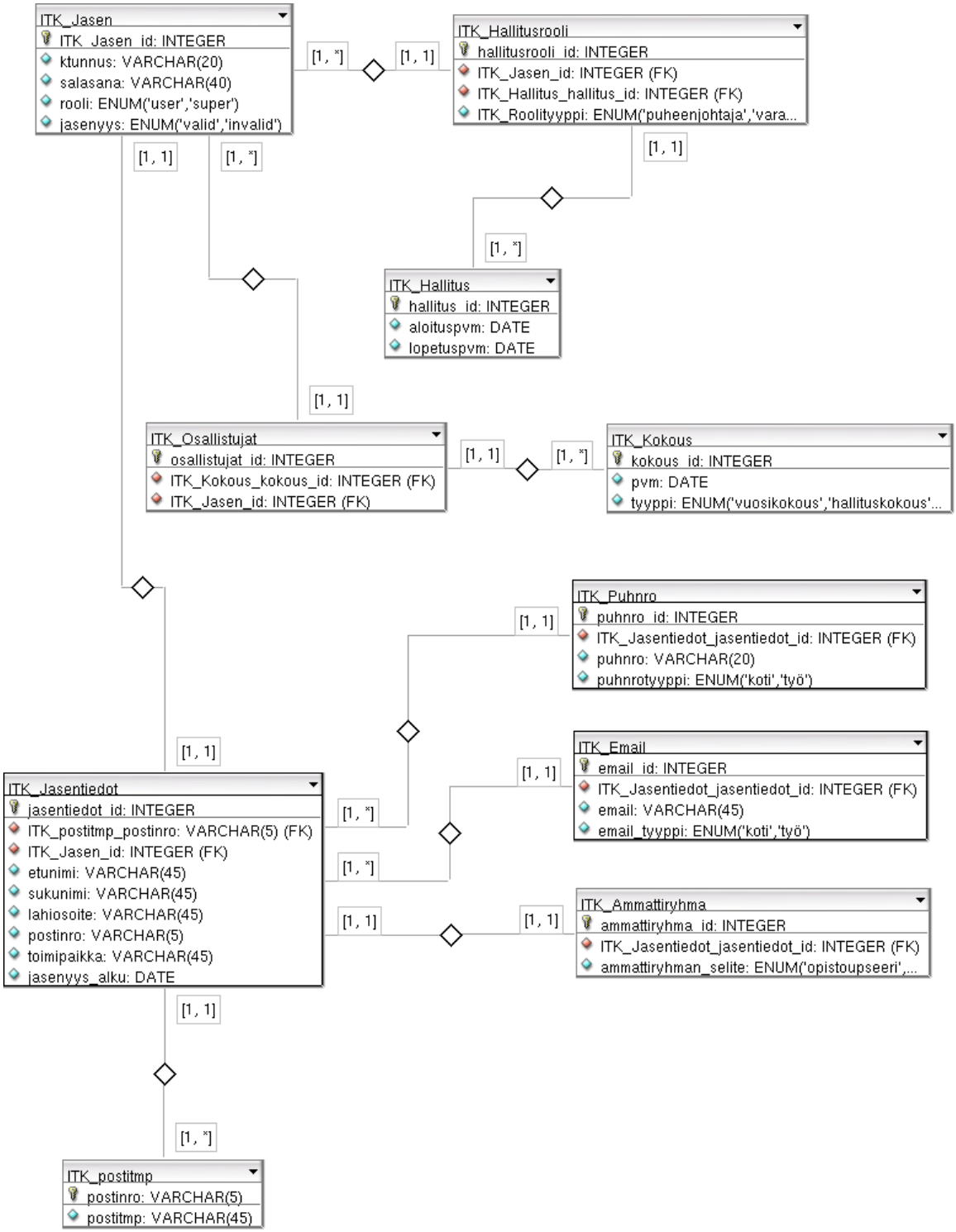
NORMAALIKÄYTTÄJÄN KÄYTTÖTAPAUSKAAVIO



PÄÄKÄYTTÄJÄN KÄYTTÖTAPAUSKAAVIO



LOPULLINEN ER-MALLI



VMWAREN NETWORKING-KONFIGUROINTITIEDOSTO

Tiedosto Muokkaa Näytä Pääte Ohje

```
VERSION=1,0
answer VNET_0_DHCP yes
answer VNET_0_DHCP_CFG_HASH 8A3C184B074BC9267C7D347FD5BE48BF9F5F939E
answer VNET_0_HOSTONLY_NETMASK 255.255.255.0
answer VNET_0_HOSTONLY_SUBNET 192.168.48.0
answer VNET_0_VIRTUAL_ADAPTER yes
answer VNET_1_DHCP no
answer VNET_1_DHCP_CFG_HASH 8A3C184B074BC9267C7D347FD5BE48BF9F5F939E
answer VNET_1_HOSTONLY_NETMASK 255.255.255.0
answer VNET_1_HOSTONLY_SUBNET 192.168.49.0
answer VNET_1_VIRTUAL_ADAPTER yes
answer VNET_2_DHCP no
answer VNET_2_DHCP_CFG_HASH 89373CFFFAC02BD67772A8669019058614CAA7C1
answer VNET_2_HOSTONLY_NETMASK 255.255.255.0
answer VNET_2_HOSTONLY_SUBNET 192.168.50.0
answer VNET_2_VIRTUAL_ADAPTER yes
answer VNET_3_DHCP no
answer VNET_3_DHCP_CFG_HASH 882B4FCEA441C55EE4901630DB02B05EBA83FEC5
answer VNET_3_HOSTONLY_NETMASK 255.255.255.0
answer VNET_3_HOSTONLY_SUBNET 192.168.51.0
answer VNET_3_VIRTUAL_ADAPTER yes
:
```

TIETOKANNAN SQL-LUONTILAUSEET

```
//Luodaan tietokanta ITK, aakkosta taulut, valitse merkistö
CREATE DATABASE ITK COLLATE utf8_general_ci;

//listaa tietokannat
SHOW DATABASES;

//valitaan käytettävä tietokanta
USE ITK;

//luodaan taulut
CREATE TABLE ITK_Jasen (
  ITK_Jasen_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ktunnus VARCHAR(20) NOT NULL,
  salasana VARCHAR(40) NOT NULL,
  rooli ENUM('user','super') NULL,
  jasenyys ENUM('valid','invalid') NULL,
  PRIMARY KEY(ITK_Jasen_id)
)
TYPE=InnoDB;

//jos haluat tarkastella
DESCRIBE ITK_Jasen;

//ITK_Hallitus-taulu
CREATE TABLE ITK_Hallitus (
  hallitus_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  aloituspvm DATE NOT NULL,
  lopetuspvm DATE NULL,
  PRIMARY KEY(hallitus_id)
)
TYPE=InnoDB;
```

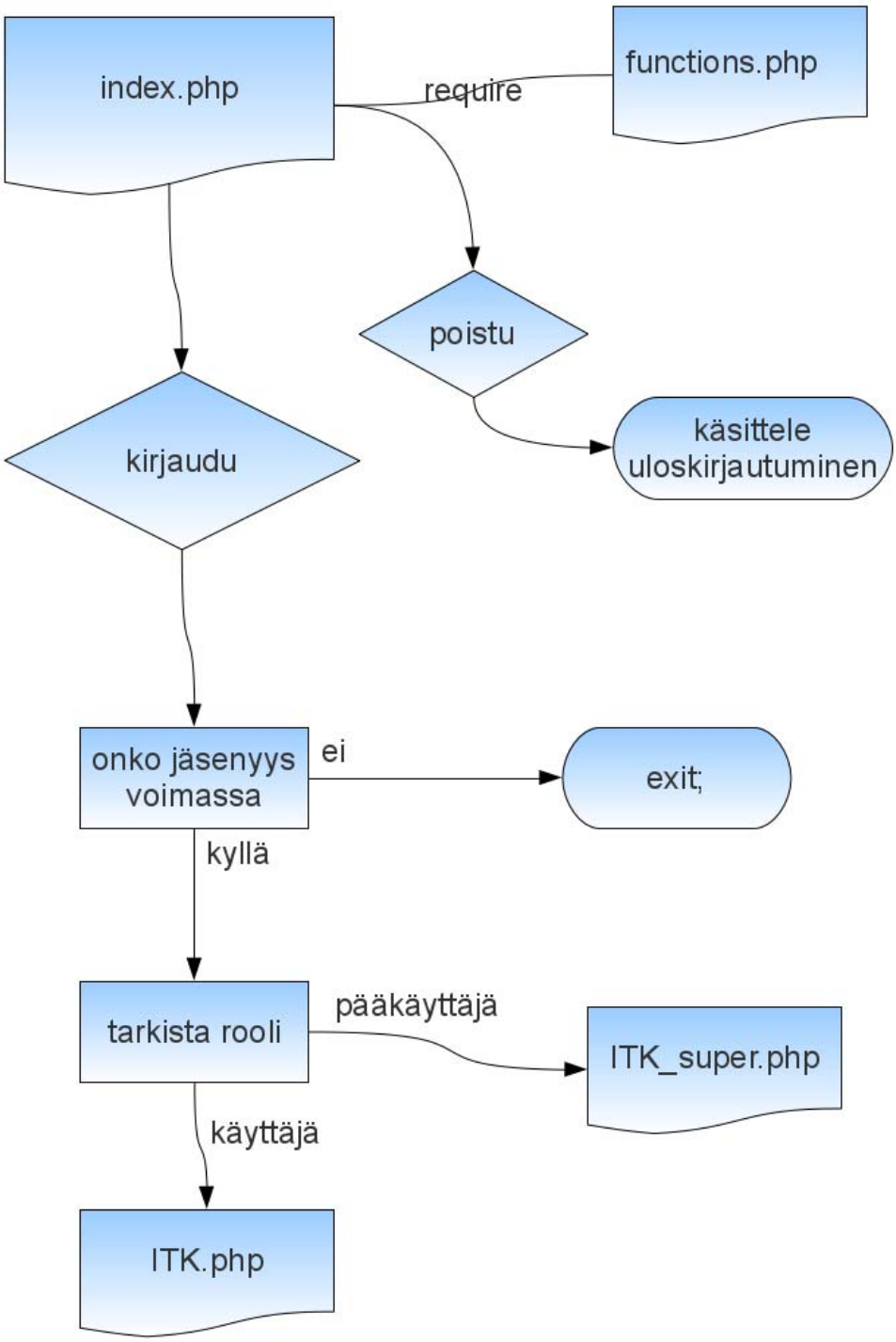
```
//ITK_Hallitusrooli
CREATE TABLE ITK_Hallitusrooli (
  hallitusrooli_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_Jasen_id INTEGER UNSIGNED NOT NULL,
  ITK_Hallitus_hallitus_id INTEGER UNSIGNED NOT NULL,
  roolityyppi ENUM ('puheenjohta-
ja','varapuheenjohtaja','sihteeri','jäsen','varajäsen','rahastonhoitaja','toiminnantarkastaja','pöytäkirja
ntarkastaja') NULL,
  PRIMARY KEY(hallitusrooli_id),
  FOREIGN KEY(ITK_Hallitus_hallitus_id)
    REFERENCES ITK_Hallitus(hallitus_id)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ITK_Jasen_id)
    REFERENCES ITK_Jasen(ITK_Jasen_id)
      ON DELETE CASCADE
      ON UPDATE NO ACTION
)
TYPE=InnoDB;

//ITK_Kokous
CREATE TABLE ITK_Kokous (
  kokous_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  pvm DATE NULL,
  tyyppi ENUM('vuosikokous','hallituskokous','perustamiskokous') NULL,
  PRIMARY KEY(kokous_id)
)
TYPE=InnoDB;

//ITK_Postitmp
CREATE TABLE ITK_Postitmp (
  postinro VARCHAR(5) NOT NULL,
  postitmp VARCHAR(45) NOT NULL,
  PRIMARY KEY(postinro)
)
TYPE=InnoDB;
```

```
//ITK_Osallistujat
CREATE TABLE ITK_Osallistujat (
  osallistujat_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_Kokous_kokous_id INTEGER UNSIGNED NOT NULL,
  ITK_Jasen_id INTEGER UNSIGNED NOT NULL,
  PRIMARY KEY(osallistujat_id),
  FOREIGN KEY(ITK_Jasen_id)
    REFERENCES ITK_Jasen(ITK_Jasen_id)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  FOREIGN KEY(ITK_Kokous_kokous_id)
    REFERENCES ITK_Kokous(kokous_id)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION
)
TYPE=InnoDB;

//ITK_Jasentiedot-taulu
CREATE TABLE ITK_Jasentiedot (
  jasentiedot_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_postitmp_postinro VARCHAR(5) NOT NULL,
  ITK_Jasen_id INTEGER UNSIGNED NOT NULL,
  etunimi VARCHAR(45) NULL,
  sukunimi VARCHAR(45) NULL,
  lahiosoite VARCHAR(45) NULL,
  toimipaikka VARCHAR(45) NULL,
  jasenyys_alku DATE NULL,
  PRIMARY KEY(jasentiedot_id),
  FOREIGN KEY(ITK_Jasen_id)
    REFERENCES ITK_Jasen(ITK_Jasen_id)
      ON DELETE CASCADE
      ON UPDATE NO ACTION,
  FOREIGN KEY(ITK_postitmp_postinro)
    REFERENCES ITK_Postitmp(postinro)
      ON DELETE NO ACTION
      ON UPDATE CASCADE
)
TYPE=InnoDB;
```

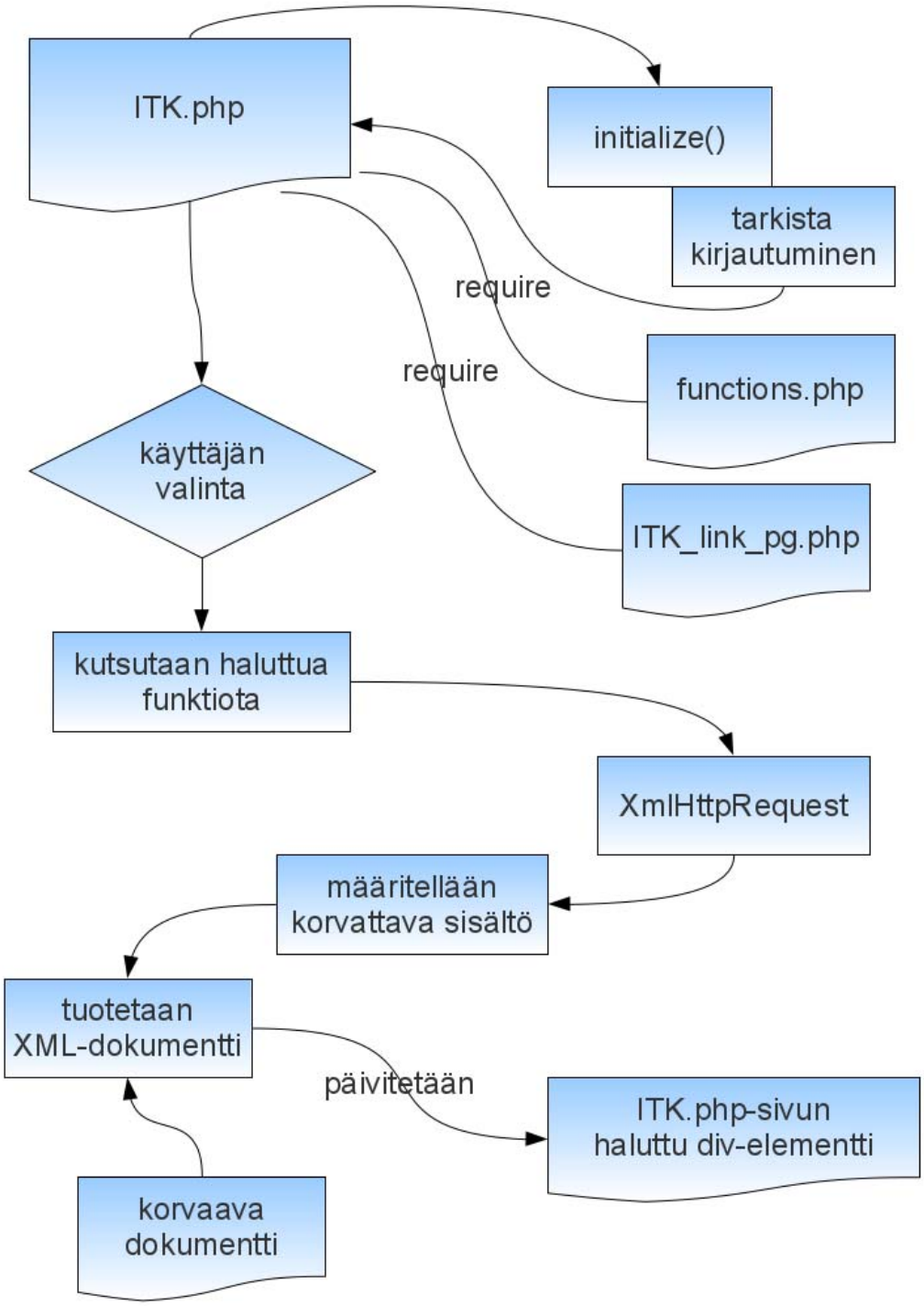
```
//ITK_Puhnro-taulu
CREATE TABLE ITK_Puhnro (
  puhnro_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_Jasentiedot_jasentiedot_id INTEGER UNSIGNED NOT NULL,
  puhnro VARCHAR(20) NULL,
  puhnrotyyppi ENUM('koti','tyo') NULL,
  PRIMARY KEY(puhnro_id),
  FOREIGN KEY(ITK_Jasentiedot_jasentiedot_id)
    REFERENCES ITK_Jasentiedot(jasentiedot_id)
      ON DELETE CASCADE
      ON UPDATE NO ACTION )
TYPE=InnoDB;

//ITK_Email-taulu
CREATE TABLE ITK_Email (
  email_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_Jasentiedot_jasentiedot_id INTEGER UNSIGNED NOT NULL,
  email VARCHAR(45) NULL,
  email_tyyppi ENUM('koti','tyo') NULL,
  PRIMARY KEY(email_id),
  FOREIGN KEY(ITK_Jasentiedot_jasentiedot_id)
    REFERENCES ITK_Jasentiedot(jasentiedot_id)
      ON DELETE CASCADE
      ON UPDATE NO ACTION )
TYPE=InnoDB;

//ITK_Ammattiryhma -taulu
CREATE TABLE ITK_Ammattiryhma (
  ammattiryhma_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  ITK_Jasentiedot_jasentiedot_id INTEGER UNSIGNED NOT NULL,
  ammattiryhman_selite ENUM('opistoupseeri','aliupseeri','upseeri','erikoisupseeri','EVP') NULL,
  PRIMARY KEY(ammattiryhma_id),
  FOREIGN KEY(ITK_Jasentiedot_jasentiedot_id)
    REFERENCES ITK_Jasentiedot(jasentiedot_id)
      ON DELETE CASCADE
      ON UPDATE NO ACTION )
TYPE=InnoDB;
```

```
//luodaan ITK_sovelluskäyttäjä ja myönnetään oikeudet
CREATE USER 'ITK_user'@'192.168.50.20' IDENTIFIED BY 'sarvikaskana';
GRANT ALL PRIVILEGES ON ITK TO 'ITK_user'@'192.168.50.20';

//luodaan testikäyttäjiä
//pääkäyttäjä: sha1-tiiviste salasanalle sarviliskoI#*T@K!TK on
//3dfdee80660dc166edfc992a7886b5c9ef0bea8b

INSERT INTO `ITK_Jasen` ( `ITK_Jasen_id` , `ktunnus` , `salasana` , `rooli` , `jasenyys` )
VALUES ('NULL', 'TeTest1', '3dfdee80660dc166edfc992a7886b5c9ef0bea8b', 'super', 'valid');
```

KIRJAUTUMISSIVUN RAKENNE

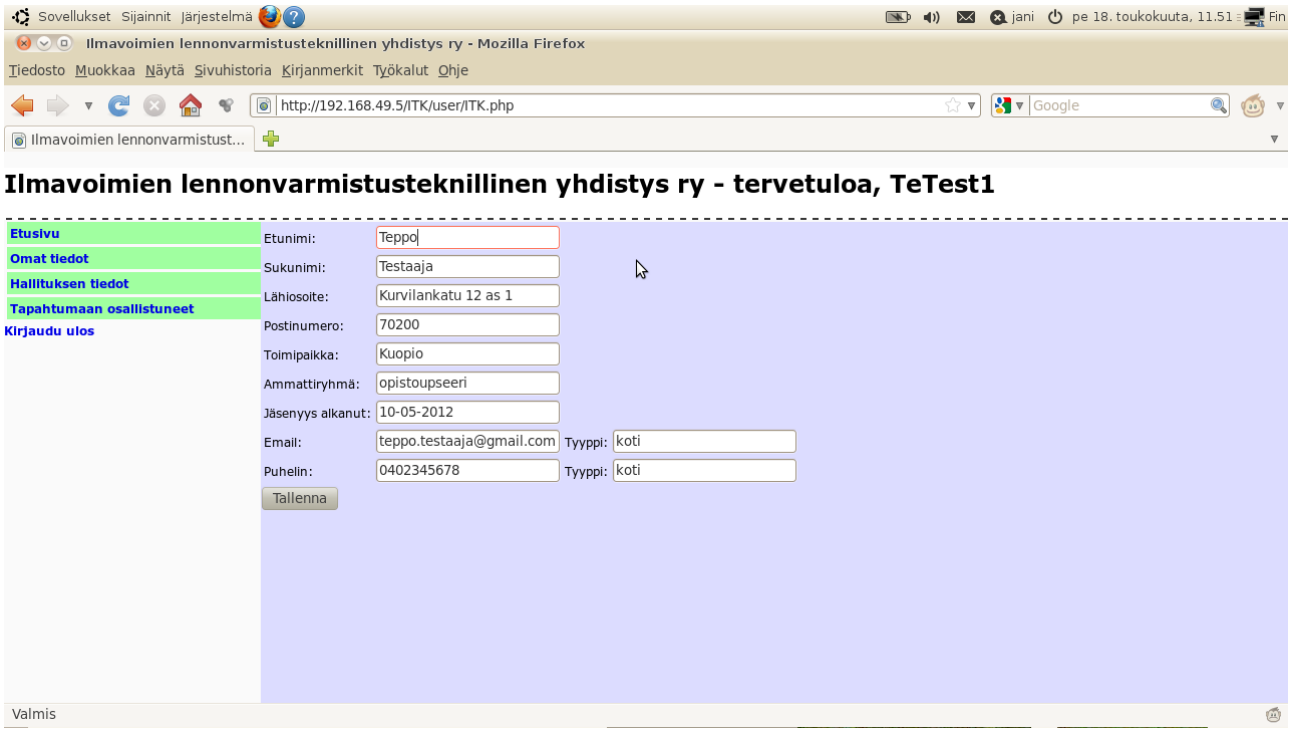


NORMAALIKÄYTTÄJÄN PÄÄSIVUN RAKENNE



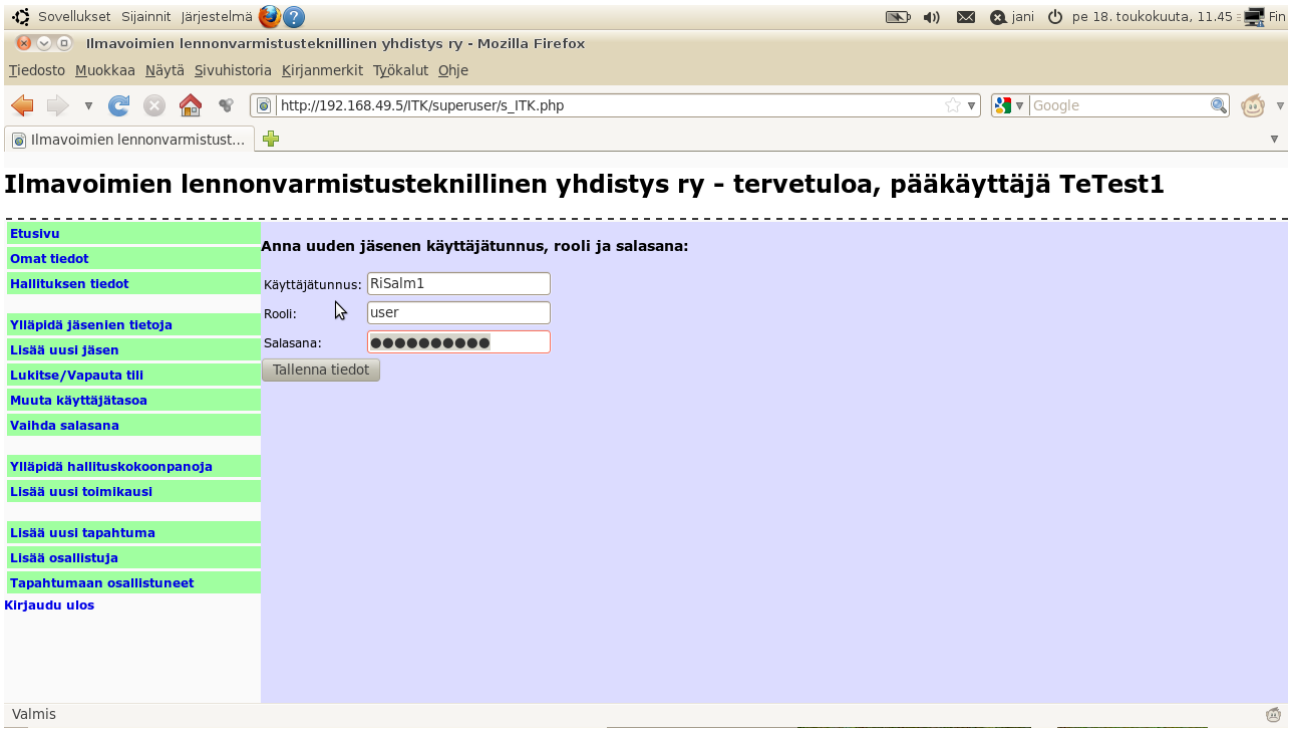
Liite 8

KÄYTTÄJÄN TIETOJEN MUUTOS -SIVU

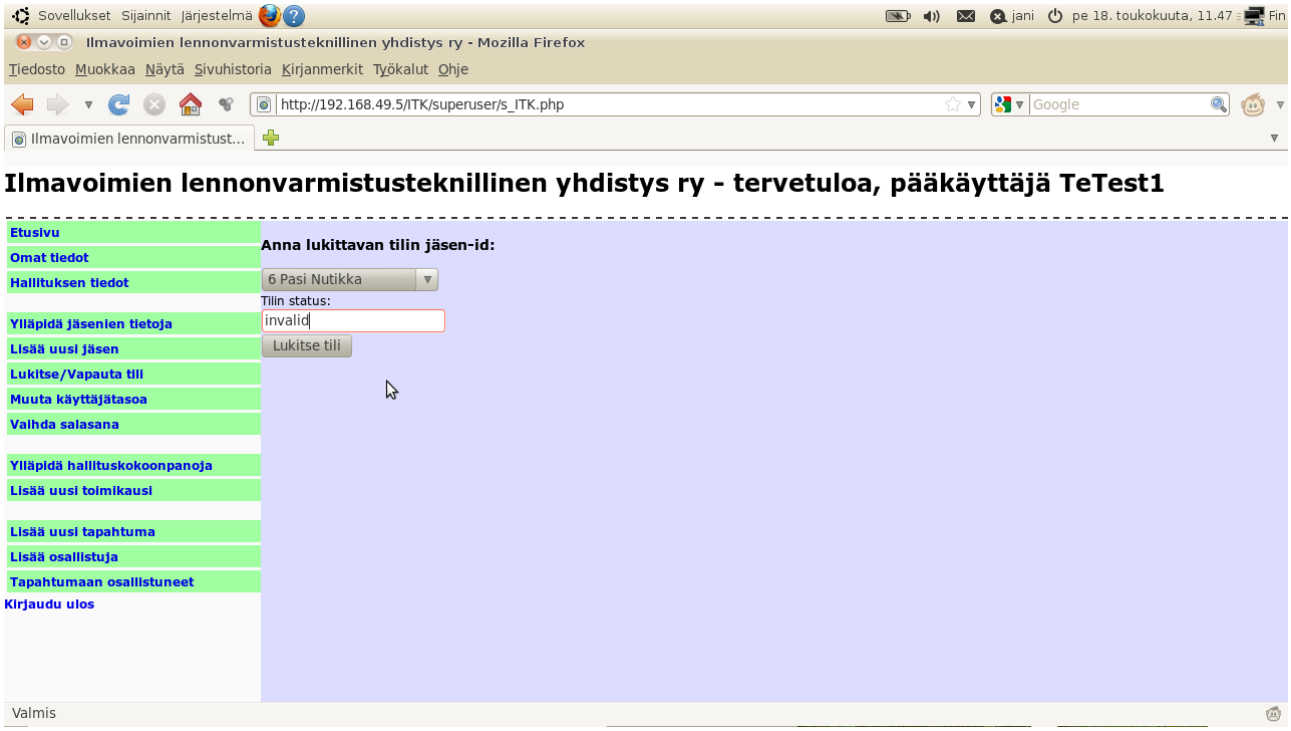


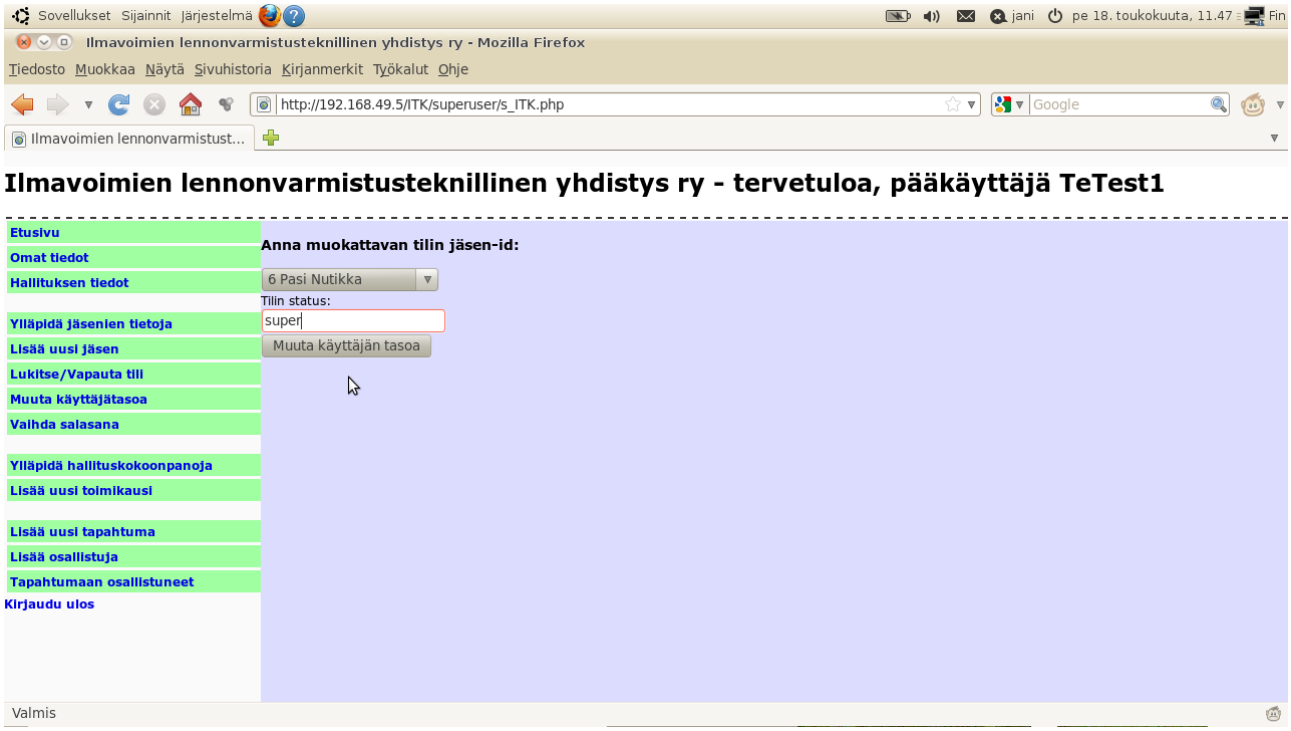
Liite 9

UUDEN JÄSENEN LISÄÄMINEN

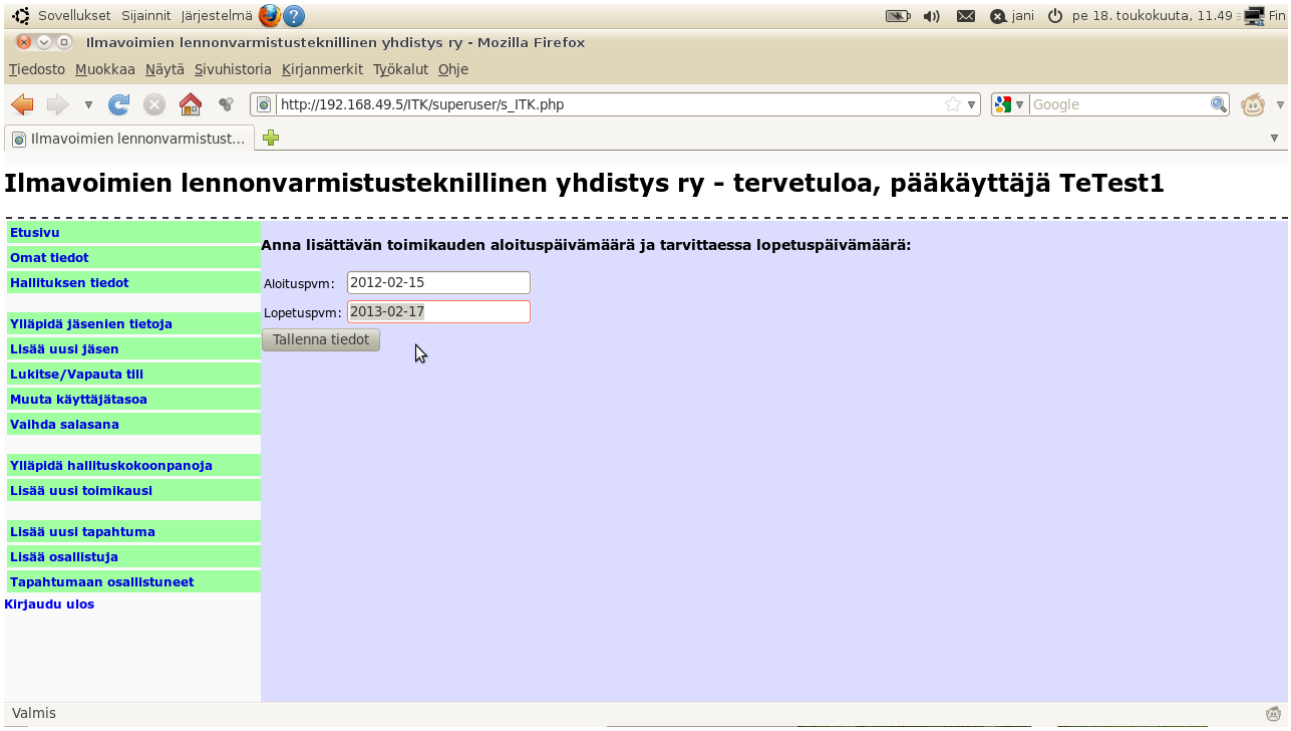


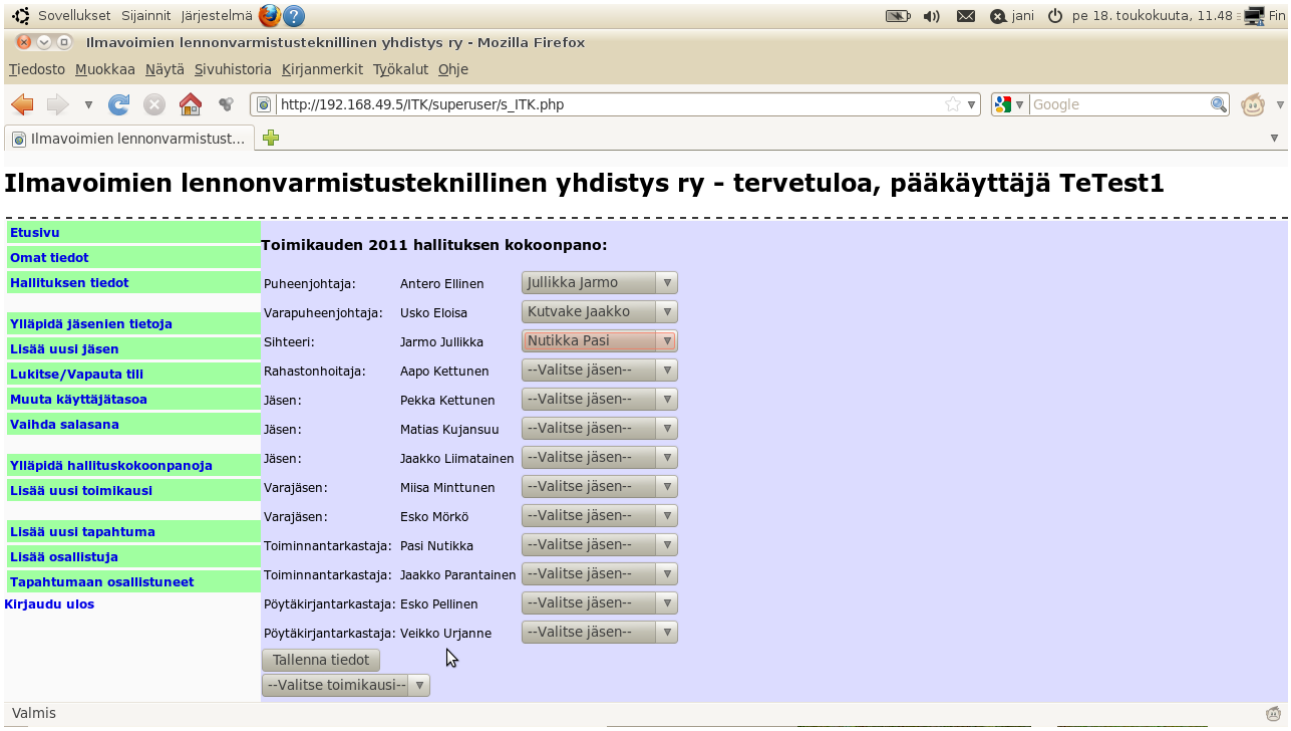
JÄSENEN TILIN LUKITSEMINEN JA KÄYTTÄJÄTASON MUUTTAMINEN



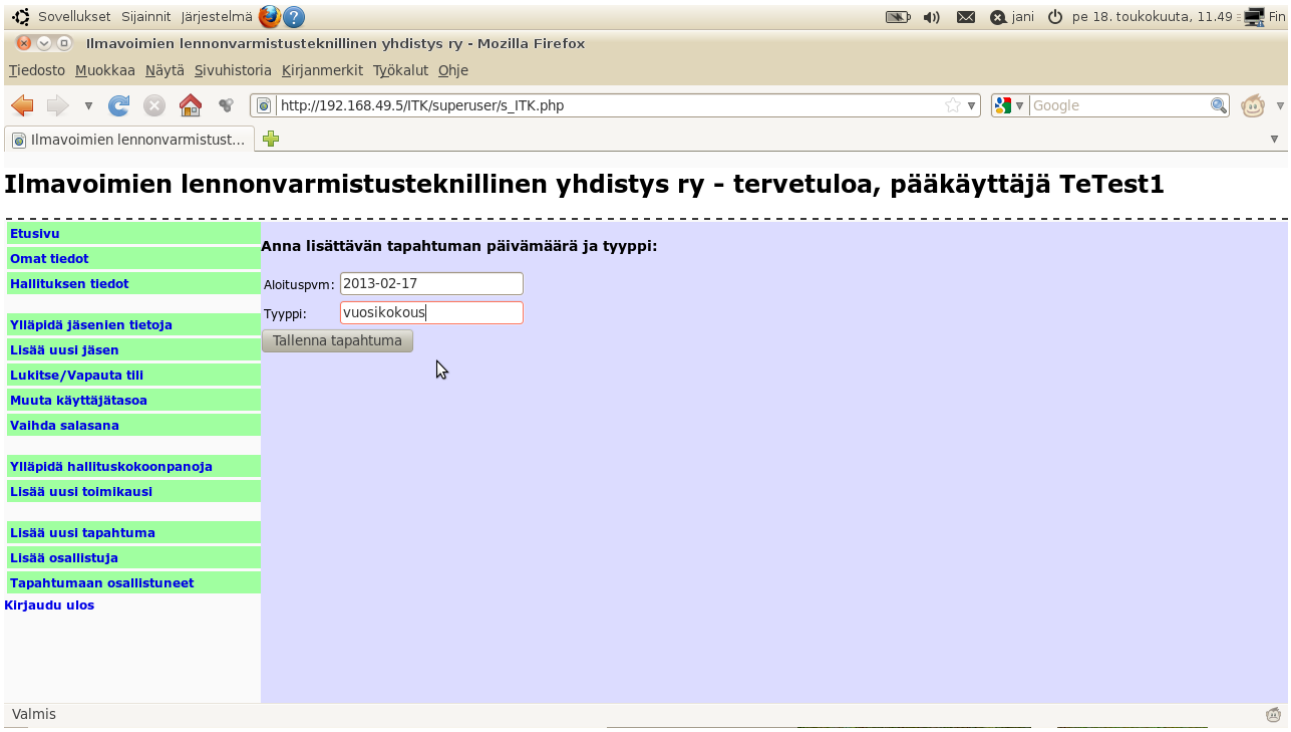


HALLITUSTIETOIHIN LIITTYVÄT PÄÄKÄYTTÄJÄN SIVUT





TAPAHTUMIIN LIITTYVÄT PÄÄKÄYTTÄJÄN SIVUT



www.savonia.fi